Proceedings

TWENTY-NINTH ANNUAL ALLERTON CONFERENCE ON COMMUNICATION, CONTROL, AND COMPUTING

October 2-4, 1991

Allerton House, Monticello, Illinois
Sponsored by the
Coordinated Science Laboratory and the
Department of Electrical and Computer Engineering of the
University of Illinois at Urbana-Champaign

PROCEEDINGS

TWENTY-NINTH ANNUAL ALLERTON CONFERENCE ON COMMUNICATION, CONTROL, AND COMPUTING

UNIVERSITY OF ILLINOIS
LIBRARY
AT URBANA - CHAMPAIGN
ENGINEERING

George Cybenko
Constatine Polychronopoulos
Mark Spong
Conference Co-Chairs

Conference held
October 2-4, 1991
Allerton House
Monticello, Illinois

Sponsored by
The Coordinated Science Laboratory
and
The Department of Electrical and Computer Engineering
of the
UNIVERSITY OF ILLINOIS
at
Urbana-Champaign

621.3811
Al5
1991

ENX

FOREWORD

Dedicated to Mac VanValkenburg, the founder of the Allerton
Conference on Communication, Control and Computing,
on the occasion of his 70th birthday.

The Twenty-Ninth Annual Allerton Conference on Communication, Control, and Computing was held at Allerton House, Monticello, Illinois on October 2, 3, and 4, 1991. The conference was sponsored by the Coordinated Science Laboratory and the Department of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign. These *Proceedings* contain papers presented at the Conference.

We are very grateful to Professor Dimitri Bertsekas of the Massachusetts Institute of Technology for his plenary lecture ''Auction Algorithms for Network Problems.'' We are also grateful to Professor Robert McEliece of the California Institute of Technology for his informative and entertaining banquet address titled ''How Shannon Saved Galileo - Maybe.'' We would like to thank all the session chairs and the authors for their contributions. Special thanks goes to the organizers of invited sessions who took time from their busy schedules to put together sessions on subjects of great current interest.

We appreciate the support of Professor W. K. Jenkins, Director of the Coordinated Science Laboratory and Professor T. N. Trick, Head of the Department of Electrical and Computer Engineering. We would like to thank our colleagues, Professors N Ahuja, K. S. Arun, P. Banerjee, M. T. Başar, R. B. Bramley, Y. Bresler, D. J. Brown, B. Hajek, T. S. Huang, W.-M. Hwu, D. L. Jones, P. R. Kumar, M. C. Loui, J. V. Medanic, S. Meyn, M. Mickunas, D. C. Munson, J. H. Patel, W. R. Perkins, J. Ponce, M. B. Pursley, D. V. Sarwate, P. M. Vaidya, and D. West for their generous assistance in reviewing papers and organizing the papers into sessions. We are also indebted to colleagues elsewhere who so willingly stepped in to assist with reviewing.

We are also grateful to Mr. Steven Williams and Ms. Susan Baxley for handling the business affairs of the Conference; to Mr. Robert Ebeling, Ms. Vicky Wolfin, and Ms. Joyce Probus for the printing of large numbers of programs and registration packets; to Mr. Jack Gladin and Mr. Bob MacFarlane for their assistance in the photography and drafting; to Ms. Beverly Becker and the wonderful staff at Allerton House who so cheerfully accommodated our group; and to Dan Jordan who handles the shipping and sales of these *Proceedings* after the Conference. Most especially we want to thank Ms. Francie Bridges, Ms. Dixie Murphy, Ms. Janet Reese, and Ms. Lila Rhoades for their year-round assistance with the Conference in handling announcements, registration, program preparation, Allerton House arrangements, and the editing of these *Proceedings*. They are the real organizers of this Conference.

George Cybenko

Constantine Polychronopoulos

Mark Spong

TABLE OF CONTENTS

FOREWORD i

I-A: MODULATION, CODING, AND EQUALIZATION FOR MOBILE RADIO COMMUNICATIONS

Chairman: D. E. Borth
University of Illinois at Urbana-Champaign

ON THE SEPARATION OF EQUALIZATION AND CODING 1
J. M. Cioffi

SOFT-OUTPUT MLSE EQUALIZATION METHODS FOR THE MOBILE RADIO CHANNEL 11
J. M. Nowack, D. E. Borth and P. D. Rasky

HIGH-CAPACITY DIGITAL CELLULAR COMMUNICATIONS THROUGH SLOW FREQUENCY-HOPPING CDMA 21
M. S. Wallace

ON THE CPM FAMILY AND ITS PATHOLOGICAL MEMBERS 31
Q. Li and B. Rimoldi

ADAPTIVE DECISION FEEDBACK EQUALIZATION TECHNIQUES FOR DIGITAL CELLULAR COMMUNICATION SYSTEMS 40
Y.-J. Liu, M. Wallace, and J. Ketchum

I-B: H_∞ AND ROBUST CONTROL

Chairman: A. S. Hodel
Auburn University

CHARACTERIZATION OF ALL CLOSED-LOOP TRANSFER FUNCTION MATRICES IN H_∞-OPTIMIZATION 50
A. A. Stoorvogel, A. Saberi and B. M. Chen

OUTPUT FEEDBACK STABLILIZING CONTROL WITH BOUNDS ON DISTURBANCE ATTENUATION 60
B. C. Fabien

H_∞ OPTIMAL CONTROL OVER A FINITE HORIZON 70
M. B. Subrahmanyam

NUMERICAL COMPUTATION OF INNER FACTORS FOR DISTRIBUTED PARAMETER SYSTEMS 72
D. S. Flamm

CLOSED-FORM SOLUTIONS TO A CLASS OF H_∞-OPTIMIZATION PROBLEMS 74
A. Saberi, B. M. Chen and Z.-L. Lin

ASSESSING THE EFFECTS OF PLANT DYNAMIC VARIATIONS ON THE ROBUSTNESS AND PERFORMANCE OF FEED FORWARD FEEDBACK 84
D. Bensoussan and M. Hammami

A TIME VARYING NEHARI THEOREM: A SYSTEM THEORETIC APPROACH 88
G. Tadmor and M. Verma

ROBUSTNESS ANALYSIS AND OPTIMIZATION........ 98
J. Kogan

TESTING ROBUST STABILITY OF GENERAL MATRIX POLYTOPES IS AN NP-HARD COMPUTATION........ 105
G. E. Coxson and C. L. DeMarco

ON STABILITY ROBUSTNESS ANALYSIS OF SYSTEMS WITH MULTIPLE LARGE PARAMETER VARIATIONS AND MULTIPLE NONLINEARITIES........ 107
M. Eslami

I-C: MATHEMATICS OF SIGNAL PROCESSING

Chairman: P. Van Dooren
University of Illinois at Urbana-Champaign

ASYMPTOTIC PROPERTIES IN ADAPTIVE FILTERING........ 117
G. Yin and L. Zhao

ROBUST LMS ADAPTIVE FILTERS FOR NONGAUSSIAN NOISE ENVIRONMENTS: CONVERGENCE PROPERTIES........ 127
G. A. Williamson and P. M. Clarkson

WAVELET DECOMPOSITION OF HARMONIZABLE RANDOM PROCESSES........ 129
P. W. Wong

2-D ZERO PHASE RATIONAL FILTERS: DESIGN AND BOUNDARY-VALUED REALIZATION........ 139
T. S. Rao and R. M. Mersereau

CONVERGENCE OF SUCCESSIVE APPROXIMATIONS FOR NONEXPANSIVE OPERATORS.... 149
L. C. Potter

FAST INVERSION OF HERMITIAN TOEPLITZ AND QUASI-TOEPLITZ MATRICES WITH ARBITRARY RANK PROFILE........ 159
D. Pal and T. Kailath

ITERATIVE ALGORITHMS FOR COMPUTING STRUCTURED RANK-REDUCING MATRIX PERTURBATIONS........ 169
M. Wicks and R. DeCarlo

SIGNAL RECONSTRUCTION OF FINITE LENGTH SEQUENCE USING INITIAL TIME SAMPLE BY HOPFIELD NEURAL NETWORK........ 179
W. Zhu, M. Zou, M. Lu and Z. Chai

I-D: INTEGRATING ROBOTICS AND COMPUTER VISION

Chairman: S. Hutchinson and J. Ponce
University of Illinois at Urbana-Champaign

FREE-FORM DEFORMABLE SURFACES FOR OBJECT MODELING AND RECOGNITION........ 181
M. Hebert, H. Delingette and K. Ikeuchi

AN ASSESSMENT OF PSEIKI FOR CAD-BASED VISION........ 191
S. G. Blask and A. C. Kak

MOBILE ROBOT MAP MAKING FROM VISION AND MOTION........ 201
D. J. Kriegman and C. J. Taylor

THREE-DIMENSIONAL RIGID-BODY POSITIONING USING CAMERA-SPACE CONTROL......... 211
U. A. Korde, W. Z. Chen, and S. B. Skaar

ANALYSIS OF DYNAMIC HAND POSITIONING TASKS USING VISUAL FEEDBACK................ 222
R. Sharma, J.-Y. Herve, and P. Cucka

SYMBOLIC ORGANIZATION OF CONTINUOUS CONTROL.. 233
R. J. Firby

I-E: FOUNDATIONS OF CONCURRENT SYSTEMS

Chairman: Gul Agha
University of Illinois at Urbana-Champaign

SCALABILITY OF PARALLEL ALGORITHMS AND ARCHITECTURES 243
V. Kumar and A. Gupta

TYPE AND SHARING ANALYSIS FOR CONCURRENT DATA STRUCTURES 253
S. Jagannathan

AN OBJECT-ORIENTED CONCURRENT REFLECTIVE ARCHITECTURE FOR DISTRIBUTED COMPUTING ENVIRONMENTS... 263
Y. Ichisugi, S. Matsuoka, T. Watanabe, and A. Yonezawa

HYPERPROGRAMS: CONCURRENT AUTOMATA AS INFORMATION STRUCTURES............... 272
P. D. Stotts and R. Furuta

PROGRAMMING WITH COMPOSITION 282
S. Taylor

II-A: FLOW CONTROL AND SCHEDULING IN NETWORKS - I

Chairman: P.R.S. Kumar
Northwestern University

ALGORITHMS FOR RESOURCE ALLOCATION IN MULTIPLE SERVICE, MULTIPLE RESOURCE COMMUNICATION NETWORKS.. 284
S. Jordan

OPTIMALITY OF INDEX POLICIES FOR STOCHASTIC SCHEDULING PROBLEMS WITH SWITCHING PENALTIES.. 294
M. P. Van Oyen, D. G. Pandelis, and D. Teneketzis

BURST-REDUCTION PROPERTIES OF LEAKY BUCKET STRATEGIES IN ATM NETWORKS... 302
V. Anantharam and P. Konstantopoulos

ON STATIONARY QUEUEING NETWORKS WITH JOB FEEDBACK 310
N. Bambos and K. Wasserman

OPTIMAL ADAPTIVE SCHEDULING IN MULTI-CLASS M/GI/1 QUEUES WITH FEEDBACK 312
P. P. Bhattacharya, L. Georgiadis, and P. Tsoucas

EFFICIENT LAYOUT OF A PASSIVE, SINGLE-HOP, FIBER-OPTIC INTERCONNECTION (AMONG N STATIONS) WITH CAPACITY $\log_2 N$.. 322
Y. Birk

ON FLOW CONTROL IN OPTICAL NETWORKS 332
R. Bolla, F. Davoli, and A. Di Febbraro

ON THE OPTIMALITY OF DECENTRALIZED THRESHOLD QUASI-STATIONARY ROUTING POLICIES IN COMPLEX DETERMINISTIC NETWORKS 342
M. Aicardi, R. Minciardi, and R. Pesenti

II-B: SPREAD SPECTRUM COMMUNICATIONS

Chairman: J. Cioffi
Stanford University

A DS/CDMA PACKET ERROR PROBABILITY APPROXIMATION 352
F. Simpson and J. M. Holtzman

ASYMPTOTIC ANALYSIS OF DIRECT-SEQUENCE SPREAD-SPECTRUM MULTIPLE-ACCESS COMMUNICATION SYSTEMS 362
J. S. Lehnert

NONCOHERENT SEQUENTIAL ACQUISITION FOR DS/SS SYSTEMS 370
S. Tantaratana and A. W. Lam

ON THE PERFORMANCE OF A TWO-STAGE ACQUISITION SCHEME FOR DIRECT-SEQUENCE SPREAD SPECTRUM 380
U. Madhow

OPTIMUM MULTIUSER DETECTION IN SINGLE-PATH RAYLEIGH FADING CHANNELS 390
Z. Zvonar and D. Brady

MULTIUSER DETECTORS FOR RICIAN FADING CDMA CHANNELS 400
S. Vasudevan and M. K. Varanasi

EXPLOITING ERROR DEPENDENCE IN A SLOW-FREQUENCY-HOPPED CDMA SYSTEM 410
M. S. McKinley and W. E. Stark

II-C: LINEAR SYSTEMS AND OPTIMAL CONTROL

Chairman: P. Voulgaris
University of Illinois at Urbana-Champaign

CANONICAL FORM OF SATURATED CONTROL OF UNSTABLE LINEAR SYSTEMS 420
G. A. Pajunen and E. I. Verriest

VARIABLE STRUCTURE OUTPUT FEEDBACK CONTROL: SWITCHING SURFACE DESIGN.. 430
R. El-Khazali and R. A. DeCarlo

REDUCED-ORDER SOLUTION FOR A CLASS OF LINEAR QUADRATIC OPTIMAL CONTROL PROBLEMS 440
D. Skataric, Z. Gajic, and D. Petkovski

ROBUST ALGORITHMS FOR TRANSFORMATION OF MIMO SYSTEMS BETWEEN CONTINUOUS- AND DISCRETE-TIME DOMAINS 448
S. Bingulac and H. F. VanLandingham

BOUNDARY VALUE PROBLEM OF LINEAR WEAKLY COUPLED SYSTEMS 455
M. T. Qureshi and Z. Gajic

SINGLE LOOP STABILITY MARGINS FOR MULTIRATE CONTROL SYSTEMS 463
D. S. Flamm

COMPUTATION OF SYSTEM ZEROS WITH BALANCING 473
A. S. Hodel and J. Tiller

II-D: GRAPH ALGORITHMS

Chairman: N. Kinnersley
University of Kansas

ON THE RECOGNITION OF STRONGLY CHORDAL GRAPHS 475
R. Sridhar and N. Chandrasekharan

AN O(NLOGN) ALGORITHM FOR FINDING A DEPTH-FIRST SEARCH TREE IN A PERMUTATION GRAPH 484
C. Rhee, Y. Liang, S. K. Dhall, and S. Lakshmivarahan

FINDING HAMILTONIAN CIRCUITS IN CIRCULAR-ARC GRAPHS 486
Y. D. Liang, C. Rhee, S. K. Dhall, and S. Lakshmivarahan

SHORTEST PATHS IN ORTHOGONAL GRAPHS 488
S. Bhatia, D. S. Hirschberg, and I. D. Scherson

BOUNDED 2 CONTRACTION OF GRAPHS 498
B. Ramamurthy and M. S. Krishnamoorthy

A PARALLEL ALGORITHM FOR FINDING MINIMUM CUTSETS IN REDUCIBLE GRAPHS 508
E. Dekel and J. Hu

MEASURING PRESORTEDNESS IN ENSEMBLES OF DATA SEQUENCES 518
R. S. Valiveti and B. J. Oommen

II-E: SIGNAL AND IMAGE ESTIMATION

Chairman: David C. Munson, Jr.
University of Illinois at Urbana-Champaign

WIDEBAND RADAR IMAGING WITH ESTIMATION THEORY APPLICATIONS 524
J. A. O'Sullivan

PERFORMANCE OF MAXIMUM LIKELIHOOD METHOD FOR DIRECTION OF ARRIVAL ESTIMATION OF WIDEBAND SIGNALS 534
R. L. Kashyap and S. M. Kattepur

ARRAY PROCESSORS BASED ON M-INTERVAL POLYNOMIAL APPROXIMATION IN TIME-DEPENDENT NOISE 543
M. Ketel and L. Kurz

A CROSS-TERM FREE JOINT TIME-FREQUENCY REPRESENTATION VIA WIGNER-VILLE DISTRIBUTION DECOMPOSITION 553
J. M. Morris and S.-E. Qian

MINIMUM VARIANCE SIGNAL ESTIMATION WITH ADAPTIVE ORDER STATISTIC FILTERS 555
P. M. Clarkson and G. A. Williamson

IMAGE SEQUENCE ENHANCEMENT USING MULTISTAGE CENTER WEIGHTED MEDIAN FILTERS 557
S.-J. Ko and T. M. Forest

TWO-STAGE ALPHA, BETA, GAMMA ESTIMATOR 559
W. D. Blair

USE OF TARGET-ORIENTED PROCESS NOISE IN TRACKING MANEUVERING TARGETS 561
W. D. Blair and J. D. Parker

JUMP-DIFFUSION PROCESSES FOR OBJECT TRACKING AND DIRECTION FINDING 563
A. Srivastava, M. I. Miller, and U. Grenander

III-A: FLOW CONTROL AND SCHEDULING IN NETWORKS - II

Chair and Organizer: B. Hajek
University of Illinois at Urbana-Champaign

BOUNDS ON MAXIMUM DELAY IN NETWORKS WITH DEFLECTION ROUTING 571
J. Brassil and R. Cruz

LARGE DEVIATIONS OF TRAFFIC IN HIGH SPEED DIGITAL NETWORKS WITH A VIEW TOWARDS NETWORK CONTROL 581
G. Kesidis and J. Walrand

A GENERALIZED PROCESSOR SHARING APPROACH TO FLOW CONTROL IN INTEGRATED SERVICES NETWORKS 591
A. K. Parekh and R. G. Gallager

BANDWIDTH AND BUFFER ALLOCATIONS IN ATM 593
S. Low and P. Varaiya

INPUT BUFFER REQUIREMENTS FOR ROUND ROBIN POLLING SYSTEMS WITH NONZERO SWITCHOVER TIMES 603
G. Sasaki

THE PROPERTIES OF NEARLY-PERIODIC TASKS AND THEIR SCHEDULING MODEL 613
S. Lee, T. Elrad, and D. Dube

ANALYSIS OF A FAIR, SLOTTED RING WITH SPATIAL REUSE 622
I. Kessler and A. Krishna

ON DYNAMIC CHANNEL ALLOCATION IN CELLULAR/WIRELESS NETWORKS 632
S.P.R. Kumar, H. W. Chung, and M. Lakshminarayan

III-B: DISTRIBUTED ALGORITHMS

Chairman: D. Muller
University of Illinois at Urbana-Champaign

EFFICIENT ALGORITHMS FOR SHARED MEMORY CONSENSUS 642
K. Abrahamson and A. Karkare

EFFICIENT CRASH RECOVERY IN SPARSE LOW DIAMETER DISTRIBUTED SYSTEMS 652
T. T.-Y. Juang and S. Venkatesan

SELF-STABILIZING MAXIMA FINDING 662
X. Lin and S. Ghosh

SELF-STABILIZING LEADER ELECTION ALGORITHMS 672
M. Flatebo and A. K. Datta

A SELF-STABILIZING ALGORITHM FOR GRAPH COLORING 674
S. Ghosh and M. H. Karaata

SEQUENTIAL CONSISTENCY AND LINEARIZABILITY: READ/WRITE OBJECTS 683
M. Mavronicolas and D. Roth

III-C: STOCHASTIC, DISCRETE EVENT AND ADAPTIVE SYSTEMS

Chairman: S. P. Meyn
University of Illinois at Urbana-Champaign

A NOVEL DISCRETE EVENT SYSTEM APPROACH TO MODELING AND ANALYSIS OF HYBRID CONTROL SYSTEMS 693
J. A. Stiver and P. J. Antsaklis

HIERARCHICAL MULTIDECISION MAKING ON A COMPUTER NETWORK WITH DISTRIBUTED COORDINATION AND CONTROL 703
J.T.C. Filho and C. P. Bottura

OPTIMAL CONTROL OF DISCRETE EVENT DYNAMICAL SYSTEMS USING NETWORK FLOW TECHNIQUES 705
R. Kumar and V. Garg

A ROBUST SMITH PREDICTOR WITH ADAPTIVE CONTROL 715
J. T. MacDonald and G. A. Williamson

ON COVARIANCE ASSIGNMENT FOR LINEAR DISCRETE-TIME SYSTEMS 717
E. Yaz and R. E. Skelton

NONPARAMETRIC DETECTION AND MEASUREMENT OF DISCONTINUITIES 727
M. Pawlak

DYNAMIC DISTURBANCE REJECTION CONTROL FOR DISCRETE NONLINEAR STOCHASTIC SYSTEMS 729
A. Azemi and E. Yaz

III-D: NEURAL NETWORKS

Chairman: A. N. Michel
University of Notre Dame

A MODEL FOR FEEDBACK NEURAL NETS WITH SUPERVISED LEARNING 731
J. Hou and F.M.A. Salam

FEEDFORWARD ARTIFICIAL NEURAL NETWORK STRUCTURE FOR CHARACTER RECOGNITION 733
M. Ahmad and F.M.A. Salam

HANDWRITTEN NUMERIC CHARACTER RECOGNITION USING A MULTILAYER NEURAL NETWORK 735
S. Benromdhane and F.M.A. Salam

NEURAL NETS WITH PRESPECIFIED MEMORIES 737
J. Aravena, M. Hegde, and M. Naraghi-Pour

LIMITATIONS OF CONSTRAINED CONNECTIVITY NEURAL NETWORKS 739
M. Hegde, M. Naraghi-Pour, and G. Nandakumar

ANALYSIS AND SYNTHESIS OF A CLASS OF DISCONTINUOUS DISCRETE-TIME NEURAL NETWORKS WITH INFINITE GAIN 741
G. Yen and A. N. Michel

HIDDEN LAYER CONFIGURATIONS OF NEURAL NETWORKS AND THEIR LEARNING PERFORMANCE 751
J.-Y. Han, X. Lei, J. Wu, and G.-Q. Chen

LEVENBERG-MARQUARDT TRAINING ALGORITHMS FOR SUPERVISED FEEDFORWARD NEURAL NETWORKS 760
J.-L. Yuan, H.-D. Chiang, C.-C. Chen, and C.-C. Yeh

IV-A: COMMUNICATION SYSTEMS

Chairman: W. Stark
University of Michigan

A PERFORMANCE ANALYSIS TECHNIQUE FOR NONLINEAR CONVOLUTIONAL/TRELLIS-CODED MODULATIONS 769
Y.-J. Liu

OPTIMUM BLOCK BASED SIGNALING OVER 1 ± D AND 1 - D^2 PARTIAL RESPONSE CHANNELS 779
A. K. Khandani and P. Kabal

PROPERTIES OF A CASCADE OF BINARY CHANNELS 790
A. B. Kiely and J. T. Coffey

UNSYMMETRICAL BOUNDARY SHAPING IN MULTIDIMENSIONAL SPACES 798
A. K. Khandani and P. Kabal

OPTIMIZING THE PERFORMANCE ON A GAUSSIAN CHANNEL WITH FINITE ALPHABET.... 809
P. F. Romain and M. Salehi

IV-B: PARALLEL ALGORITHMS

Chairman: D. Brown
University of Illinois at Urbana-Champaign

FUNDAMENTAL ALGORITHMS ON RECONFIGURABLE MESHES 811
S. Olariu, J. L. Schwing, and J. Zhang

INTEGER PROBLEMS ON RECONFIGURABLE MESHES, WITH APPLICATIONS 821
S. Olariu, J. L. Schwing, and J. Zhang

PARALLEL CIRCULAR-ARC GRAPH ALGORITHMS ON THE HYPERCUBE 831
M. G. Andrews and D. T. Lee

A HYPERCUBE ALGORITHM FOR THE MAXIMA PROBLEM 841
S.-Q. Zheng and F. Lee

SIMPLE AND EFFICIENT IN-PLACE PARALLEL SORTING ALGORITHMS 843
Y. Zhang and S.-Q. Zheng

A PARALLEL ALGORITHM FOR THE CLASS OF $\pm 2^b$ ASCEND COMPUTATIONS ON A SIMD HYPERCUBE 845
D. Nassimi

SOME DOUBLY LOGARITHMIC TIME OPTIMAL PARALLEL ALGORITHMS 847
Z. Wen

PARALLEL ALGORITHMS FOR ARITHMETIC EXPRESSION PARSING 856
W. Deng and S. S. Iyengar

IV-C: ROBUST CONTROL

Chairman: Bassam Bamieh
University of Illinois at Urbana-Champaign

ON PARAMETRIC ROBUST STABILITY AND PERFORMANCE 866
M. Dahleh, A. Tesi and A. Vicino

CONTINUITY PROPERTIES OF THE REAL/COMPLEX STRUCTURED SINGULAR VALUE 875
A. Packard and P. Pandey

SYSTEMS WITH STRUCTURED UNCERTAINTY: RELATIONS BETWEEN QUADRATIC AND ROBUST STABILITY 885
M. Rotea, M. Corless, D. Da, and I. R. Petersen

ROBUSTNESS ANALYSIS OF NONLINEAR SYSTEMS 895
J. S. Shamma

FINITE HORIZON H_∞ DIGITAL CONTROL FOR SAMPLED-DATA SYSTEMS 905
K. M. Nagpal, W. Sun, and P. P. Khargonekar

APPROXIMATELY OPTIMAL FINITE DIMENSIONAL CONTROLLERS IN THE 2-BLOCK H^∞ PROBLEM FOR DISTRIBUTED PLANTS 912
H. Ozbay

IV-D: IMPLEMENTATION ISSUES IN SIGNAL PROCESSING

Chairman: D. Pal
AT&T Bell Labs

OPTIMAL BI-LEVEL QUANTIZATION 922
J. Fu, S. Tang and K. Zhang

METHODS FOR APPROXIMATING THE SHORT-TIME ENERGY OF NOISE-CORRUPTED SPEECH SIGNALS 925
S. A. Dimino, R. J. Niederjohn, J. A. Heinen, and T. J. Svoren

PREFILTERING AND SELECTIVE LINEAR PREDICTION TO IMPROVE FORMANT TRACKING OF NOISE-CORRUPTED SPEECH 927
T. J. Svoren, R. J. Niederjohn, J. A. Heinen, and R. B. Wallace

CONCURRENT IMPLEMENTATION OF DIGITAL FILTERS 929
J. L. Aravena and N. Bhattacharyya

RESPONSE CLASSIFICATION OF FLOATING POINT REALIZATIONS OF LINEAR DISCRETE SYSTEMS 939
P. H. Bauer and J. Wang

THE INFLUENCE OF QUANTIZATION FORMATS ON ABSOLUTE ERROR BOUNDS IN FLOATING POINT AUTOREGRESSIVE MODELS 948
P. H. Bauer and J. Wang

IV-E: QUANTIZATION

Chairman: S. Tantaratana
University of Massachusetts

ADAPTIVE TRELLIS AND DPCM CODING OF COMPOSITE PROCESSES 958
M. Naraghi-Pour, M. Hegde, S. Das, and N. Arora

OPTIMAL CYCLIC QUANTIZATION ON CLOSED PEANO CURVES 968
X. Wu

UNEQUAL ERROR PROTECTION (UEP) CODES APPLIED TO TRANSMISSION OF ANALOGUE SOURCES 977
H. Arda Aksu and M. Salehi

THE ANALYSIS OF THE C-CHAIN CODE 979
Y.-W. Chen and S. C. Lee

PLENARY LECTURE

AUCTION ALGORITHMS FOR NETWORK FLOW PROBLEMS: A TUTORIAL INTRODUCTION 989
Dimitri P. Bertsekas

V-A: COMPUTATIONAL GEOMETRY AND VLSI

Chairman: M. Sarrafzadeh
Northwestern University

GENERALIZED INTERSECTION SEARCHING PROBLEMS 1003
R. Janardan and M. Lopez

ON THE EFFICIENT RECOGNITION OF THE ELLIPSE 1013
Y. C. Cheng and S. C. Lee

2-CONCAVE RECTILINEAR POLYGONS—NECESSARY AND SUFFICIENT FOR GRAPH DUALIZATION 1023
K. H. Yeap and M. Sarrafzadeh

LINEAR PLACEMENT OF GRAPHS WITH A MINIMUM PATH COVER 1033
P.-Y.F. Lin and K. Nakajima

DESIGNING MULTI-LINK ROBOT ARMS IN A CONVEX POLYGON 1043
I. Suzuki and M. Yamashita

V-B: SEMICONDUCTOR MANUFACTURING SYSTEMS

Chairman: Shu Li
Intel Corporation

MANAGEMENT OF SEMICONDUCTOR MANUFACTURING USING DISCRETE EVENT SIMULATION 1053
W. Grey and S. Hood

THE PROGRAMMABLE FACTORY 1061
R. E. Null

EQUI-VARIABILITY GRAPH APPROACH FOR MODELING OF MANUFACTURING SYSTEMS 1063
S. Li

STATISTICAL CIRCUIT DESIGN CENTERING USING RESPONSE SURFACE METHODOLOGY 1071
M. R. Rencher

V-C: SCHEDULING AND ROUTING APPLICATIONS IN COMPUTER SYSTEMS

Chairman: R. Cruz
University of California at San Diego

PEBBLER SHIFT WITH PRESPECIFIED PATHS 1081
J. Narasimhan, C. S. Rim, and K. Nakajima

IMPROVED GLOBAL ROUTING λ-GEOMETRY 1083
S. Burman, H. Chen and N. Sherwani

DESIGN OF MULTIPLE BUS INTERCONNECTION NETWORKS FOR FAN-IN COMPUTATIONS 1093
R. Vaidyanathan

SHORTCUT ROUTING IN DEADLOCK-FREE NETWORKS 1103
F.C.M. Lau and S. W. Lau

A GENERALIZED DIMENSION EXCHANGE SCHEME FOR DYNAMIC LOAD BALANCING IN DISTRIBUTED MEMORY MULTIPROCESSORS 1105
C. Z. Xu and F.C.M. Lau

V-D: NONLINEAR CONTROL AND ROBOTICS

Chairman: M. Spong
University of Illinois at Urbana-Champaign

ROBUSTNESS IN NONLINEAR FEEDBACK SYSTEMS 1114
D. Bensoussan

BIPED CONTROL VIA NONLINEAR DYNAMICS 1117
H. M. Hmam and D. A. Lawrence

THE PHASE PORTRAIT ANALYSIS OF THE JOINT DRIFT MOTION FOR KINEMATICALLY REDUNDANT ROBOTS 1127
S. Luo and S. Ahmad

ADAPTIVE CONTROL OF MULTIPLE ROBOTS IN COOPERATIVE MOTION 1137
M. Zribi and S. Ahmad

INDEX OF AUTHORS 1147

On the Separation of Equalization and Coding

John M. Cioffi*
Information Systems Laboratory, Stanford, CA 94305 415-723-2150

October 1, 1991

Abstract

Combined design of the coding and equalization functions for data transmission on linear dispersive additive Gaussian noise channels can lead to performance very close to fundamental theoretical limits, as reviewed herein. We investigate the situation when the code and equalization functions are **separately** optimized and show that the combination of equalization and coding can remain optimized in this case. We illustrate two examples of such separately optimized systems, coded multicarrier modulation and coded single-carrier modulation with decision feedback. We also interpret the results in terms of broadcast digital mobile data transmission.

Contents

1 Introduction 1
- 1.1 ISI Channel Model 2
- 1.2 Traditional Approach 2

2 Capacity and Multicarrier Modulation 3
- 2.1 Mutual information and capacity for the ISI channel 3
  - 2.1.1 The ISI-free AWGN channel 3
  - 2.1.2 Calculating Capacity for the ISI channel 4
- 2.2 Discrete Multicarrier Modulation 5

3 Single-Carrier Modulation 6

4 Discussion of Wireless (Time-Varying) Case 8

5 Conclusion 9

1 Introduction

The problem of high-speed data transmission on channels with **intersymbol interference** (ISI) has received special recent attention under the topical classification of **combined equalization and coding.** A variety of combined equalization and coding methods have been recently devised that permit reliable data transmission at rates close to fundamental theoretical limits (i.e., near capacity). Of these, the two most popular are coded multicarrier modulation [1], [2], and trellis-coded/precoded decision-feedback methods [3], [4]. In restrospect, both of these approaches can be found to be largely separable in the equalization and coding functions, as we discuss in this paper. Combined equalization and coding methods are now finding their way into a number of wired transmission applications such as voiceband modems, [1], [5] and high-speed

---

*This work was supported by an NSF PYI Award

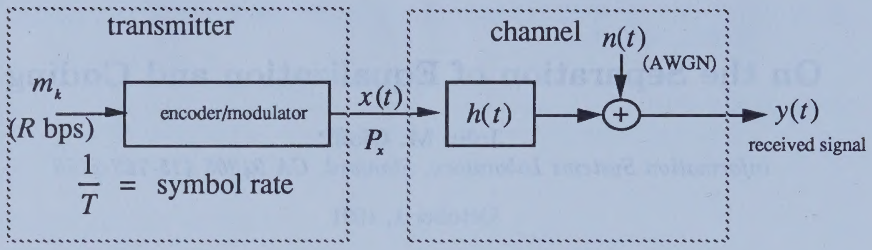


Figure 1: Linear intersymbol interference channel with additive Gaussian noise.

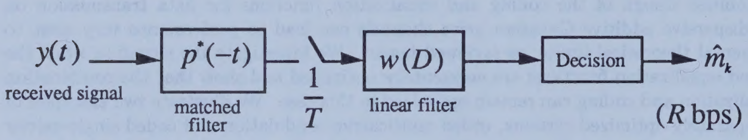


Figure 2: Traditional linear equalizer for intersymbol interference channel.

digital transmission on subscriber loops [6], [7], [8]. However, it is in data transmission over wireless time-varying channels where the separation has most signficance. In this paper, we illustrate the concept of separation of equalization and coding in general terms for multicarrier and for single-carrier modulation with trellis coding in Sections 2 and 3, respectively. We then comment on the meaning of this separation in the context of broadcast data transmission with mobile receivers in Section 4.

1.1 ISI Channel Model

This paper's focus is on the linear, dispersive additive Gaussian noise channel in Figure 1. A data bit stream of R bits per second (bps) is grouped into $b = RT$ bits per **symbol period** T to form the message m_k at time kT. The message is encoded into a continuous-time modulated signal $x(t)$ that is filtered by a channel with impulse response $h(t)$, before white Gaussian noise of (two-sided) power spectral density σ^2 is added.[1] The received signal is $y(t) = h(t) * x(t) + n(t)$. The **transmit power** is constrained and denoted P_x, so that the **energy per symbol** is $\mathcal{E}_x = P_x T$. The rate $1/T$ is often called the **symbol rate**.

A channel that can be modelled as in Figure 1 is characterized by a data rate C called the **capacity**. For any rate $R < C$, there exist sets of modulated signals such that when the received signal is decoded with a maximum likelihood decoder, the **probability of symbol error**, P_e, will be arbitrarily small. It is the study and determination of the structure of these signal sets that motivates this paper and combined equalization and coding.

1.2 Traditional Approach

The traditional approach to such an ISI channel had previously been to use a structure, known as a **linear equalizer**, shown in Figure 2 to make the channel appear as if it were ISI-free at the equalizer output. The structure consists of a matched filter, $p^*(-t)$ ($p(t) = h(t) * \varphi(t)$, and $\varphi(t)$ encompasses any and all transmit filtering), symbol-rate sampling, and then a symbol-rate linear filter that attempts to minimize the mean-square difference between the filter output and the transmitted signal $x(t)$ or its discrete equivalent. Good codes, convolutional ($b < 1$ bit/dimension) or trellis ($b \geq 1$ bits/dimension), designed for the non-dispersive additive white Gaussian noise channel were then concatenated to improve performance. The matched filter output is information lossless and the discrete time filter is invertible, if designed correctly, so that the receiver processing is information lossless and an appropriate decoder would be able to decode any transmitted message with minimum P_e. However, such an encoder would be of prohibitive complexity in practice, so the decoding is nominally performed as if there were no dispersion in the channel. That is, a decoder for only the applied code is used in the receiver and the channel effects and equalizer are ignored. Further, the code used is not such that performance would be close to capacity even if the maximum-likelihood decoder could be constructed. Thus, a code designed for an ISI-free channel could be used and peformance is improved by the gain of that code. This approach often leads to a maximum reliably transmitted data rates well below capacity. Combined equalization and coding methods entail the joint design of equalization and coding functions to increase the achievable data rates. Of these, coded multicarrier and coded decision-feedback methods are two of the best known forms of combined equalization and coding methods.

---

[1] No generality is lost in assuming additive white noise in that noise with shaped power spectrum can always be transformed into an equivalent channel model of the form of Figure 1.

2 Capacity and Multicarrier Modulation

We briefly review the calculation of capacity for the intersymbol interference channel of Figure 1 and show how one specific form of multicarrier modulation derives directly from this calculation. We then proceed to show that the coding and equalization functions in the resultant structure can be interpretted as having been separately optimized.

2.1 Mutual information and capacity for the ISI channel

We begin by reviewing the concept of mutual information and capacity for the ISI-free channel and then proceed to use the reviewed concepts for the channel with ISI. Then, we investigate capacity for the channel with intersymbol inteference as an aggregate of ISI-free channels.

2.1.1 The ISI-free AWGN channel

For a complex channel with $h(t) = \|h\|^2\delta(t)$ and normalized transmit filter ($\int_{-\infty}^{\infty} |\varphi(t)|^2 dt = 1$), the capacity is well known to be [9]

$$C(T) = \log_2\left(1 + SNR\|h\|^2\right) \text{ bits/symbol} \quad , \tag{1}$$

where $SNR = \mathcal{E}_x/2\sigma^2$.[2] In bits/second, the capacity and mutual information are computed as $C = C(T) \cdot \frac{1}{T}$. By random coding arguments, we know that a large number of good codes exist such that P_e can be made very small for any rate $R < C$. For any given code, or set of signals for $x(t)$, we often define, at some fixed P_e (say 10^{-6}), a quantity known as the **SNR gap**, Γ [5]. Γ measures the loss in SNR, or equivalently in data rate with respect to capacity. Using Γ, the achievable data rate is

$$b = \log_2\left(\frac{1 + SNR\|h\|^2}{\Gamma}\right) = C - G \quad , \tag{2}$$

---

[2] Note that since we have a complex channel, which is inherently two real dimensions, the capacity formula does not have the usual factor of 1/2 in front, and the SNR has the complex signal energy divided by the complex noise variance, the latter of which is $2\sigma^2$. One can identically analyze the system in terms of real dimensions using the factor of 1/2 and the same SNR.

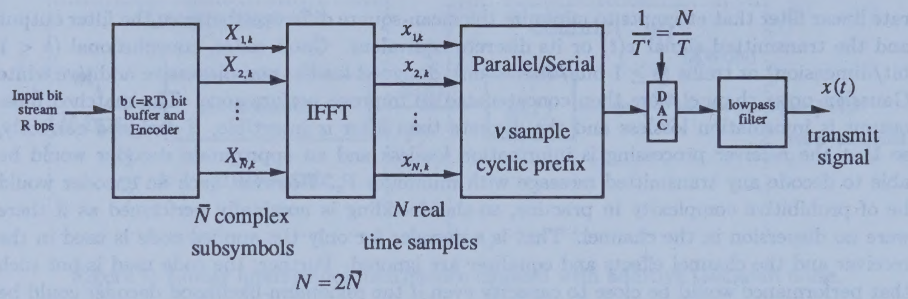


Figure 3: Transmitter for Discrete Multitone (DMT) modulation.

where $G = \log_2(\Gamma)$ is the **bit-rate gap**. Good codes exist for both the cases $b \geq 1$ [10], [11], [12] and $b < 1$ [13] so that at $P_e = 10^{-6}$, the bit-rate gap is less than 1 bit/symbol or very small, respectively.

Thus, fundamentally, coding for the ISI-free channel is well understood and theoretical limits can be closely approached in practice for sufficiently complex implementations. Perhaps not surprisingly in retrospect, it will be these same good codes, designed without knowledge of intersymbol interference, that are also good for the channel of Figure 1 when the equalization function has been separately optimized with only the knowledge that one of these good codes is to be used. While this result seems to have been anticipated in antiquity, it is only recently that it has been used in a productive sense to design very high performance transmission systems.

2.1.2 Calculating Capacity for the ISI channel

When the channel is dispersive $h(t) \neq \|h\|^2\delta(t)$, and we can divide the transmit bandwidth $[0, \frac{1}{2T'})$ into $\bar{N}$ (let $\bar{N}$ be even) contiguous non-overlapping frequency **bins**, each of width $1/T$ where $T = NT'$ and $N = 2\bar{N}$. These bins have center frequencies that are integer multiples of $1/T$ (the zero frequency bin and the Nyquist frequency bin are combined and considered as one bin equivalent in bandwidth to all the others). Let these subchannels be indexed by $i = 1, ..., \bar{N}$ with $i = \bar{N}$ corresponding to the combined DC-Nyquist bin. For sufficiently large $\bar{N}$, these slots will be roughly modelled by independent ISI-free channels, each having capacity as given in (1), with $\|h\|^2$ replaced by $|H_i|^2$, the value of the channel transfer function at the i^{th} center frequency. For fixed power budget, each channel has an energy $\mathcal{E}_i$ and an $SNR_i = \mathcal{E}_i/(2\sigma^2)$. The mutual information (not capacity unless the $\mathcal{E}_i$ are optimized jointly) for the aggregate set of independent bins is

$$I(T) = \sum_{i=1}^{\bar{N}} \log_2\left(1 + SNR_i \cdot |H_i|^2\right) \text{ bits/symbol }, \tag{3}$$

with $H_i \triangleq H(i/T), i = 1, ..., \bar{N} - 1$ and $H_{\bar{N}} = H(0) + \jmath H(1/2T')$. The capacity is found by using the famous water-pouring solution [14] for the energy allocation for SNR_i in 3:

$$SNR_i + \frac{1}{|H_i|^2} = \text{constant} \quad , \tag{4}$$

where the energy must be nonnegative and $\mathcal{E}_x = \sum_{i=1}^{\bar{N}} \mathcal{E}_i$. The mutual information and capacity in bits/second are then $I = I(T)/T$ and $C = C(T)/T$, respectively. By taking limits as $T \to 0$,

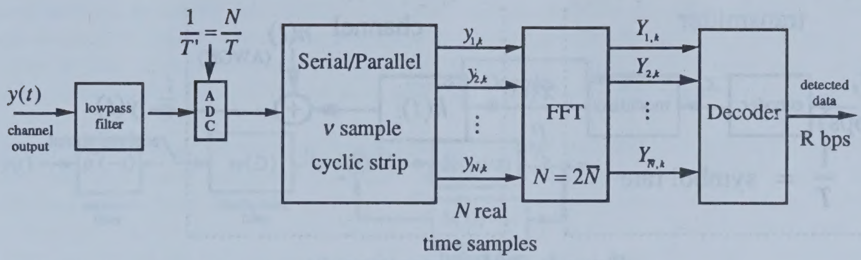


Figure 4: Receiver for Discrete Multitone (DMT) modulation.

one determines the two formulae:

$$I = \int_{\Omega} \log_2 \left(1 + SNR(f)|H(f)|^2\right) df \tag{5}$$

$$C = \int_{\Omega^*} \log_2 \left(1 + SNR(f)|H(f)|^2\right) df \tag{6}$$

where $SNR(f) = \mathcal{E}(f)/\sigma^2$ and Ω^* is the limit of the frequency interval corresponding to the nonzero part of the water-pouring energy distribution, while Ω is the interval for any chosen (nonzero) energy distribution, for energy distribution $\mathcal{E}(f)$.

The question addressed in combined equalization and coding is then how to design a code that will achieve close to these theoretical limits, with the development above suggesting multicarrier modulation.

2.2 Discrete Multicarrier Modulation

Discrete Multitone (DMT) modulation [15], [2] is a direct implementation of the structure suggested by capacity and/or mutual information calculation in (3) using a digital implementation of the modulation/demodulation functions by Discrete Fourier Transforms (most often implemented as Fast Fourier Transforms (FFTs)). The transmitter for DMT is shown in Figure 3. For finite $\bar{N}$ in practice, each transmit block must be cyclically prefixed by ν samples, where $\nu T'$ is the approximate duration of the channel pulse response $p(t)$. With this prefix, the subchannels derived by the corresponding receiver in Figure 4 are independent and have the gains H_i as listed in the capacity derivation above. Since each of these subchannels are ISI-free, then those same good codes that were used on the ISI-free channel can be used on each of the DMT subchannels.[3] The reliabily transmitted number of bits per symbol with a specific code (code with same gap on all subchannels) would then be

$$b = \sum_i b_i \tag{7}$$

$$= \sum_i \log_2 \left(\frac{1 + SNR_i |H_i|^2}{\Gamma} \right) \tag{8}$$

$$= \log_2 \prod_i \left(\frac{1 + SNR_i |H_i|^2}{\Gamma} \right) . \tag{9}$$

---

[3] In practice, see [16], the same code is used sequentially on several or all the subchannels rather than using a separate code on each subchannel.

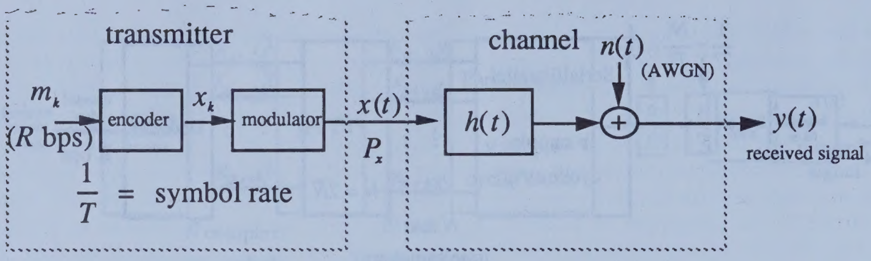


Figure 5: Single-carrier transmitter.

The form of (9) suggests an equivalent SNR for comparison with an ISI-free channel, with symbol rate altered to $1/T'$ as

$$\overline{SNR\|h\|^2} \triangleq \left[\prod_i \left(1+SNR_i|H_i|^2\right)\right]^{1/\bar{N}} . \tag{10}$$

The products, sums, and $\bar{N}$ above are understood to represent only those subchannels that are used (i.e., those with nonzero transmit energy $\mathcal{E}_i > 0$). Then we see, that

$$b = \bar{N}\log_2\left(\frac{1+\overline{SNR\|h\|^2}}{\Gamma}\right) , \tag{11}$$

which we see is an equivalent ISI-free type result for the reliably transmitted data rate and a gap characterized by the same code for an ISI-free channel.

In practice, one finds that a flat energy distribution, but over the same band as dictated by water-pouring (or close to it), is all that is needed to get very close to capacity with a good code and small gap. Nevertheless, even if this band of optimum frequencies were known a priori without explict identification of the channel characteristic in detail at the transmitter, DMT would still need to know the channel characteristic at the channel so that the $b_i = \log_2\left(\frac{1+SNR_i|H_i|^2}{\Gamma}\right)$ could be computed for application of the code. Nevertheless, we see a separation of the equalization function, which encompases the division of the channel into parallel independent subchannels through the use of FFT's and cyclic prefix, and the application of codes otherwise known to be good for any ISI-free channel. In so doing, we can see through (10) and (11) that the resultant bit rate with this separation of equalization and coding functions is as close to capacity (same gap) as would be achieved on an equivalent ISI-free channel with the same code, same capacity, and no intersymbol inteference. Thus, by separation of the equalization and coding function, DMT is a means by which to support near optimum data transmission on any channel of the form of Figure 1. On the other hand, the need for knowing b_i, which depends heavily on the channel characteristic, in the actual implementation of the code does imply some "combination" of the equalization and coding functions. We shall see that this allocation of bits is somewhat artificial, and can be circumvented, in the sections to follow.

3 Single-Carrier Modulation

While the multicarrier approach is optimum (for large $\bar{N}$), there is another approach that is more along the traditional lines of Figure 2. Surprisingly, this approach can achieve the same data rates and does so with a detector that is not maximum likelihood, but only a close approximation to maximum likelihood.

The transmitter of Figure 1 is redrawn for what is known as **single-carrier** modulation in Figure 5. The messages are converted at the symbol rate to (complex) scalar **symbols**, x_k, which are in turn modulated with filter $\varphi(t)$, and applied to the channel. The equalizer

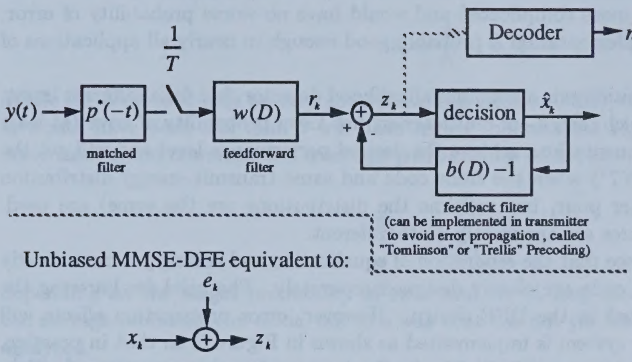


Figure 6: Decision feedback (DFE) receiver.

of Figure 2 is replaced by the Decision Feedback Equalizer (DFE) receiver of Figure 6. The matched filter output is converted to a minimum-phase signal (in the mean-square error sense) by the filter $W(D)$, and previous decisions are used to construct with the filter $1 - B(D)$ and then to eliminate through subtraction any trailing intersymbol interference in that resultant minimum-phase equivalent signal. The optimum settings for the feedforward filter, $W(D)$, and feeback filter $B(D)$ in Figure 6 are found via spectral factorization as described in [3]. When these filters are appropriately chosen, the DFE decision-element input sequence z_k is equivalent to the additive noise channel also shown in Figure 6, where x_k is again the transmitted symbol and e_k is an error added to x_k. e_k depends on the noise $n(t)$ and also on other (future) values of the input symbol sequence x_{k+i}, $i \geq 1$, but is independent (at time k only) of the symbol x_k. The error e_k corresponds to an unbiased receiver and so is not quite white, although it is usually very close to white [3]. An observation first made in [3] is that the mutual information in 5 is related to the SNR of this equivalent channel, $\mathrm{SNR}_{\mathrm{MMSE\text{-}DFE,U}} \triangleq \mathcal{E}_x/\sigma_e^2$, through the relation:

$$I(T) = \log_2(1 + \mathrm{SNR}_{\mathrm{MMSE\text{-}DFE,U}}) \quad . \tag{12}$$

Furthermore, when the transmit filter $\varphi(t)$ is chosen to have a spectral shaping that is the same as the water-pour energy distribution, then that

$$C(T) = \log_2(1 + \mathrm{SNR}_{\mathrm{MMSE\text{-}DFE,U}}) \quad . \tag{13}$$

The relation in (13) is the same as that for an ISI-free channel. The fact that e_k is not quite white is not of consequence in coding because interleaving methods can always be used to avoid projection of correlated noise in a destructive manner onto minimum distance error events. Thus, a good code (with gap Γ) could be used on this channel, and the achievable bit rate would then be

$$b = \log_2\left(\frac{1 + \mathrm{SNR}_{\mathrm{MMSE\text{-}DFE,U}}}{\Gamma}\right) = C - G \quad . \tag{14}$$

The better the code, the smaller the gap, and the closer that x_k appears to a Gaussian sequence, so then e_k looks almost exactly Gaussian also. Further, we know from the converse to Shannon's famous coding theorem that the system could not be further improved, as we can come arbitrarily close to $b = C(T)$ with this DFE-equivalent channel, and any further improvement could not lead to reliable transmission at a higher rate. Any receiver decoding need only be implemented for the applied code, and need not know what the channel characteristics are, when the DFE is used. Nevertheless, the receiver is not a maximum-likelihood receiver (but apparently close enough so that P_e goes to zero also when a good code is used). The maximum likelihood receiver would be more complicated and would have no worse probability of error, but is unnecessary as this implementation is probably good enough in nearly all applications of current interest.

The multicarrier implementation is a maximum likelihood detector, but for a different input, so it is not possible to say which (single- or multicarrier) has lower probability of error (at least easily). However, both implementations achieve the desired performance level and will get the same data rate (when $T = NT'$) when the same code and same transmit energy distribution (not necessarily just for water pour, but anytime the distributions are the same) are used. Nevertheless, the two structures are apparently quite different.

In the DFE case, we also see that the separation of equalization and coding yet more clearly as the DFE and the applied code are clearly designed separately. The need for knowing the b_i of multicarrier is eliminated in the DFE design. However, error propagation effects will be severe if the single-carrier system is implemented as shown in Figure 6, so that in practice, precoding methods [5] are used to move $B(D)$ (the feedback section) to the transmitter.[4]. This, in practice, requires channel identification and knowledge of the channel in detailed sense (i.e., more than just a good guess at the optimum bandwidth to use) in the transmitter. Thus, in practice (at least as far as we've discussed so far in this paper), both DMT and the DFE need to know the channel characteristics, in detail, in the transmitter to transmit data at the highest possible limits.

Fundamentally, one looks at the DFE system where we see that $\mathrm{SNR}_{\mathrm{MMSE\text{-}DFE,U}}$ is equivalent to $\overline{SNR}$ in the DMT system, and that with infinite complexity (or no error propagation), the DFE system would not need detailed knowledge of the channel at the transmitter. This, then, raises the question as to whether the multicarrier system needs to know the channel in detail, or whether this is a function of the particular embodiment of the DMT system that has forced this knowledge. The answer is apparently the latter, and has led, increasingly, to use of multicarrier in time-varying mobile channels as we summarize in Section 4.

4 Discussion of Wireless (Time-Varying) Case

In time-varying mobile data transmission, it is usually not possible to know the exact channel characteristic at the transmitter. This is because the time the receiver requires to estimate the channel and communicate the result to the transmitter exceeds the rate of significant variation of the channel in most practical situations. While the codes used in the combined equalization and coding methods described earlier in this paper were independent of the channel characteristic (with no loss in performance), some part of the equalization function always required detailed knowledge of the channel in the transmitter. Thus, the designer is faced with an apparent quandry in that to get maximum performance, it appears the channel need be known in detail to the transmitter. Analytical and experimental evidence is beginning to accumulate that the channel knowledge is not necessary for near-maximum performance. The DFE without error propagation would have been one good combined equalization and coding method that could have circumvented the need for accurate channel knowledge, but practical constraints currently prevent implementation of this method.

The DMT system previously derived specfically required that the bit allocation, b_i, be known in the transmitter. The b_i, in turn, depend on detailed knowledge of the channel. However, recall that our development of DMT relied on an assumption that all subchannels use a code with the same gap Γ. This assumption tacitly implies that we desire an equal error probability, P_e, on each subchannel. This is usually a constraint imposed in multicarrier, but we instead now remove this constraint and allow the coded subchannels to have different error rates with the same code and gap, by forcing the b_i to be constant such that

---

[4] Other methods known as "decision feedback sequence detection" or "reduced state sequence detection" have been proposed to eliminate error propagation in the receiver, but they are too complicated to implement in practice

$$b_i = \frac{b}{\bar{N}} = \bar{b} \ . \tag{15}$$

While the code used may be very good (have gap close to 0 dB), some of the subchannels where b_i would have been less than $\bar{b}$ will have higher-than-desired $P_{e,i}$. Furthermore these higher error rates could dominate the **average probability of error**,

$$\bar{P}_e = \frac{1}{\bar{N}} \sum_{i=1}^{\bar{N}} P_{e,i} \ , \tag{16}$$

depending on the target probability of error and $\bar{N}$. A long block length $\bar{N}$ can help reduce the average probability of error, but in a way that has not yet been completely understood nor analyzed.

We therefore only comment here on two emerging, nonexclusive, methods that can be used to achieve a target $\bar{P}_e$. The first is called **tone shuffling** [8], which mixes the order index of the tones in both the transmitter and correspondingly in the receiver in observation of the fact that subchannels with $b_i < \bar{b}$ are likely to be adjacent or close. Then, the applied gain of any trellis code will likely be applied only to one (or a few) subchannels with apparently large noise within an error event. The trellis code can then improve these subchannels basically by pulling down the performance of some of the good subchannels to help those with power distance to noise ratio. This techniques was also used to fight bandlimited noise and impulse noise in subscriber loops in [8].

The second method is to apply error correcting codes to the new equal-b_i multicarrier modulation, thereby sacrificing a small amount of rate but taking advantage of the robustness of some error correcting methods to burst errors. Elements of this method appear in some of the emerging technologies for digital replacement of the FM radio band in North America, although the connections to combined equalization and coding discussed here do not seem apparent to any of the authors of the proposed solutions. For a sampling of one specific version of such an equal-bit multicarrier, see the work by Wozencraft and Moose of Mercury Digital Communications [17] among the many that have been proposed.

5 Conclusion

Combined equalization and coding methods can be used to achieve reliable transmission of the highest possible data rates on channels with intersymbol interference. Further, the equalization and coding functions can be largely separated so that good codes known for the ISI-free channel can also be used (with the right kind of equalization) on channels with intersymbol interference with no performance loss. In a sense, this result is dual to Shannon's third theorem stating that optimal compression and optimal coding can be separated. Finally, it appears that the separation of equalization function from code design can be exploited to avoid the need for detailed channel knowledge at the transmitter, particularly in the multicarrier case, thus indeed leaving multicarrier modulation as a very interesting method for wireless data transmission on time-varying mobile channels.

References

[1] J.A.C. Bingham. "Multicarrier Modulation for Data Transmission: An Idea Whose Time has Come". *IEEE Communications Magazine*, 28(4):5–14, April 1990.

[2] A. Ruiz, J.M. Cioffi, and S. Kasturia. "Discrete Multiple Tone Modulation with Coset Coding for the Spectrally Shaped Channel". *IEEE Transactions on Communications*, to appear 1991.

[3] J.M. Cioffi, G.P. Dudevoir, M. V. Eyuboglu, and G.D. Forney. "MMSE Decision-Feedback Equalizers and Coding – Part I: General Results". submitted, IEEE transactions on Communications, 5/90.

[4] M.V. Eyuboglu and G.D. Forney. "Trellis Precoding". submitted, *IEEE Transactions on Information Theory*. See also, *Proceedings, ARO Communications Workshop*, Ruidoso, NM, May 1989. See also *Proceedings, IEEE International Symposium on Information Theory*, San Diego, January 1990.

[5] G.D. Forney Jr. and M.V. Eyuboglu. "Combined Equalization and Coding using Precoding". *IEEE Communications Magazine*, 29(12), December 1991.

[6] I. Kalet and N. Zervos. "Single-tone and Multi-tone: The Connection". In *International Conference on Communication*, Boston, June 1989.

[7] P.S. Chow, J. Tu, and J.M. Cioffi. "Performance Evaluation of a Multichannel Transceiver System for ADSL and VHDSL Services". *IEEE Journal on Selected Areas in Communication*, 8(4):??, August 1991.

[8] J.S. Chow, J. Tu, and J.M. Cioffi. "A Discrete Multitone Transceiver System for HDSL Applications". *IEEE Journal on Selected Areas in Communication*, 8(4):??, August 1991.

[9] R.E. Blahut. *Principles and Practice of Information Theory.* Addison-Wesley, Menlo Park, CA, 1987.

[10] G. Ungerboeck. "Channel Coding with Multilevel/Phase Signals". *IEEE trans. on Information Theory*, 28(1):55–67, January 1982.

[11] G.D. Forney Jr. "Coset Codes - Part I: Introduction and Geometrical Classification". *IEEE Transactions on Information Theory*, 34(5):1123–1151, September 1988.

[12] G.D. Forney Jr. "Trellis Constellations". *IEEE transactions on Information Theory*, submitted 1989.

[13] Shu Lin and Jr. Daniel J. Costello. *"Error Control Coding, Fundamentals and Applications"*. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1983.

[14] R.G. Gallager. *"Information Theory and Reliable Communication"*. Wiley, New York, NY, 1968.

[15] A. Peled and A. Ruiz. "Frequency Domain Data Transmission using Reduced Computational Complexity Algorithms". In *International Conference on Acoustics, Speech, and Signal Processing*, April 1980. Denver, pp. 964-967.

[16] A. Ruiz and J.M. Cioffi. "A Frequency-Domain Approach to Combined Spectral Shaping and Coding". In *ICC'87*, pages 1711–1715, Seattle, WA, June 1987. to appear, IEEE com. trans.

[17] J.M. Wozencraft and P.H. Moose. "Modulation and Coding for In-Band Digital Audio Broadcast using Multi-Frequency Modulation". In *National Association of Broadcasters 1991 Conference Proceedings, see also Radio 1991 Proceedings (San Francisco, 9/91)*, Las Vegas, April 1991.

Soft-Output MLSE Equalization Methods for the Mobile Radio Channel

Joseph M. Nowack, David E. Borth and Phillip D. Rasky

Communication Systems Research Laboratory
Corporate Research & Development Center
Motorola, Inc.
1301 E. Algonquin Road
Schaumburg, IL 60196

Abstract

Soft-decision decoding with equalization is a challenging problem in high-speed digital communications over the land mobile radio channel. In many systems, a maximum-likelihood sequence estimator (MLSE) using the Viterbi algorithm is used in the receiver as a channel equalizer to mitigate the effect of intersymbol interference (ISI). In coded systems, the MLSE equalizer precedes the decoder and should provide the decoder with soft information concerning the equalizer output symbols to take advantage of the coding gains attainable with soft-decision decoding. This paper addresses the performance and complexity of three different methods of generating the soft information: the metric difference method, the ISI-cancellation method and the Soft-Output Viterbi Algorithm. Simulation results are given for a mobile radiotelephone system operating on a land mobile radio channel where the channel model includes multipath fading. Applications of these methods include mobile radio systems using convolutional codes or trellis coded modulation.

1 System Description

Assume that in a mobile radio system, a maximum-likelihood sequence estimator (MLSE) channel equalizer precedes a decoder to estimate the transmitted sequence (see Figure 1). Soft information associated with each MLSE output symbol can be determined by performing a symbol-by-symbol maximum *a posteriori* probability (MAP) decoding of the received sequence [1]. Although this method minimizes the symbol error probability for each symbol given the entire received sequence, it is impractical for many applications due to the computational complexity and excessive storage requirements. Instead, several pragmatic methods have been developed such as the *metric difference method*, the *ISI-cancellation method* and the *Soft-Output Viterbi Algorithm* [2] which can be implemented with only minor modifications of the standard MLSE structure.

From [3], the MLSE algorithm finds the most likely transmitted sequence over an intersymbol interference channel by maximizing the survivor metric $J_n(\sigma_n)$ for each state

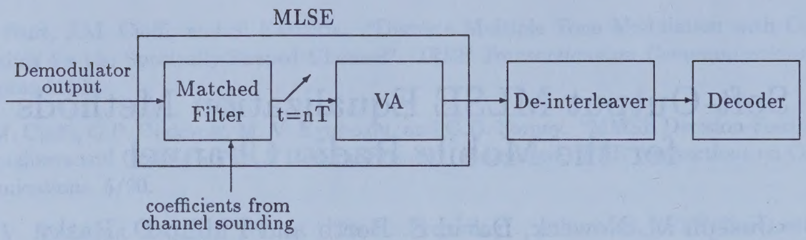


Figure 1: Receiver with maximum-likelihood sequence estimator and decoder.

σ_n at time n via the operation on the matched filter output z_n,

$$J_n(\sigma_n) = 2 \text{ Re } (\alpha_n^* z_n) + \max_{\{\sigma_{n-1}\}\to\sigma_n} \{J_{n-1}(\sigma_{n-1}) - F(\sigma_{n-1}, \sigma_n)\}, \tag{1}$$

where the maximum is taken over the set of all states $\{\sigma_{n-1}\}$ that have σ_n as a possible successor state and where

$$F(\sigma_{n-1}, \sigma_n) = \alpha_n^* s_0 \alpha_n + 2 \text{ Re } \left(\alpha_n^* \sum_{l=1}^{L} s_l \alpha_{n-l}\right), \tag{2}$$

where L is the channel memory in bits; s_i, $0 \leq i \leq L$, is a function of the channel impulse response; and α_{n-l} is the hypothesized transmitted data symbol at time $t = nT$. A state σ_n is defined as the data sequence $\sigma_n = \alpha_{n-L+1}, \alpha_{n-L+2}, \ldots, \alpha_n$; thus every state sequence is related to a specific data sequence.

In this paper, a differentially-decoded binary GMSK system having transmitted data values $\alpha_i = \{\pm 1, \pm j\}$ is considered. A maximum channel memory $L = 4$ is assumed thus requiring a 16-state MLSE equalizer. Letting bit $u_i = 0$ correspond to $\alpha_i = +1$ or $+j$ and letting $u_i = 1$ correspond to $\alpha_i = -1$ or $-j$, each state σ_n can be written as a binary 4-tuple,

$$\sigma_n = (u_{n-3} u_{n-2} u_{n-1} u_n), \tag{3}$$

where $u_i = 0$ or 1 depending on the value of α_i. The data is assumed to be framed (≈ 60 bits) and channel state information is available for the MLSE from channel soundings.

2 Soft-Output MLSE Equalization Methods

2.1 Metric Difference Method

For the system of interest, the maximization process in (1) over the set of all possible sequences σ_{n-1} at time $n-1$ leading to state σ_n at time n is just over the two states $\sigma_{n-1} = (u_{n-4} u_{n-3} u_{n-2} u_{n-1})$ which differ in the most significant bit u_{n-4}, i.e., the maximization process is over the two states $\sigma_{n-1} = (0xyz)$ or $\sigma_{n-1} = (1xyz)$ which lead to the state

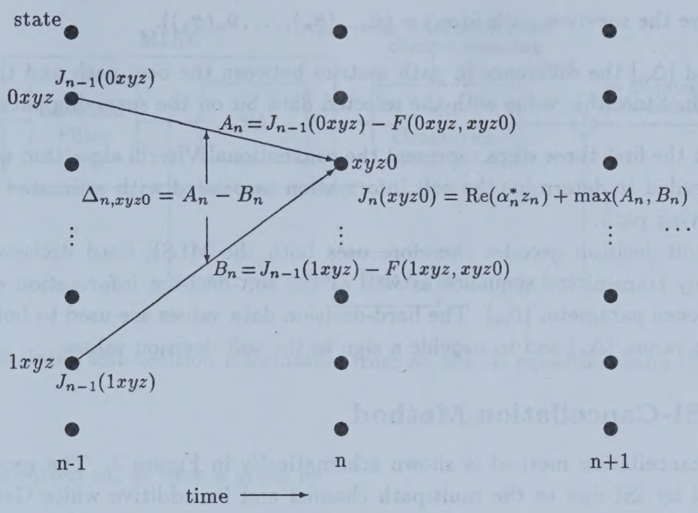

state
$J_{n-1}(0xyz)$
$0xyz$
$A_n = J_{n-1}(0xyz) - F(0xyz, xyz0)$
$xyz0$
$\Delta_{n,xyz0} = A_n - B_n$
$J_n(xyz0) = \mathrm{Re}(\alpha_n^* z_n) + \max(A_n, B_n)$
$B_n = J_{n-1}(1xyz) - F(1xyz, xyz0)$
$1xyz$
$J_{n-1}(1xyz)$
n-1
n
n+1
time

Figure 2: Path metric difference at one state in MLSE trellis.

$\sigma_n = (xyzw)$ where w, x, y and z may be 0 or 1. Figure 2 shows a segment of the corresponding trellis diagram in which two states at time $n-1$ have branches leading to state $\sigma_n = (xyz0)$ at time n. The survivor metric is for the path that maximizes

$$\max_{\{\sigma_{n-1}\}\to\sigma_n} \{J_{n-1}(\sigma_{n-1}) - F(\sigma_{n-1}, \sigma_n)\}. \tag{4}$$

Let $A_n = J_{n-1}(0xyz) - F(0xyz, xyz0)$ and let $B_n = J_{n-1}(1xyz) - F(1xyz, xyz0)$. Then, the upper or lower branch is selected as the survivor path depending upon the sign of the difference $\Delta_n = A_n - B_n$. That is, if $\Delta_n > 0$, the path from $\sigma_{n-1} = 0xyz$ is selected; otherwise the path from $\sigma_{n-1} = 1xyz$ is selected. The absolute value of Δ_n is a measure of the difference between the two choices of sub-sequences $0xyz$ and $1xyz$ [4]. If $|\Delta_n|$ is small, then there is a high probability that the signal sequence was noisy and that the bit chosen at time n (bit u_{n-4}) may be in error. Note that Δ_n accounts for both

- signal level variations in α_{n-4} through the dependence of Δ_n on the matched filter outputs $z_{n-4}, z_{n-3}, z_{n-2}, z_{n-1}$; and,
- intersymbol effects on α_{n-4} through the dependence of Δ_n on $\alpha_{n-3}, \alpha_{n-2}, \alpha_{n-1}$, and α_n; i.e., $|\Delta_n|$ is a measure of the equalized symbol signal quality.

The metric difference method [5] can be summarized as follows. For each state σ_n at time n:

1. Compute the path metrics for the paths leading to state σ_n.
2. Compare the path metrics, select the best path and store the metric.

3. Store the survivor path $\hat{u}(\sigma_n) = (\hat{u}_{n-\tau}(\sigma_n), \ldots, \hat{u}_n(\sigma_n))$.

4. Find $|\Delta_n|$ the difference in path metrics between the best path and the next best path. Store this value with the selected data bit on the surviving path $\hat{u}(\sigma_n)$.

Note that the first three steps represent the conventional Viterbi algorithm while the last step is needed to determine the soft information associated with estimated bit $\hat{u}_{n-L}$ on the surviving path.

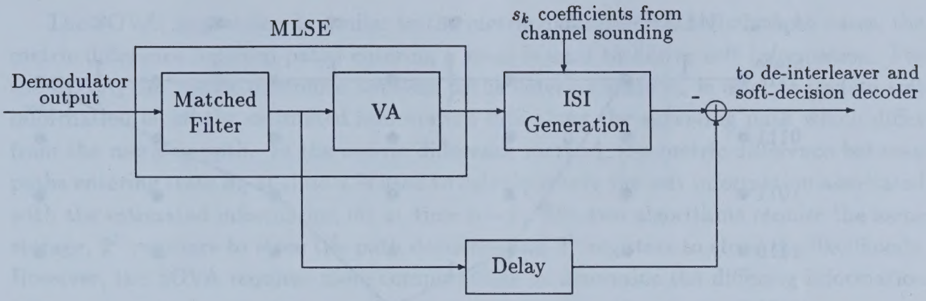
The soft-decision decoder therefore uses both the MLSE hard decision about the most likely transmitted sequence as well as the soft-decision information contained in the difference parameter $|\Delta_n|$. The hard-decision data values are used to both index the difference values $|\Delta_n|$ and to provide a sign to the soft decision values.

2.2 ISI-Cancellation Method

The ISI-cancellation method is shown schematically in Figure 3. The received signal, corrupted by ISI due to the multipath channel and by additive white Gaussian noise (AWGN) due to the receiver front end, is equalized using an MLSE equalizer. The MLSE equalizer employs a matched filter (MF) that is matched to the multipath channel, a Viterbi algorithm and two data conversion units. For the system of interest, the output of a quadrature demodulator is fed to the MF section of the MLSE. A channel sounding process is used to determine the coefficients for this matched filter. The complex output is then converted to a real sequence since only real values are ultimately sent to the decoder. The complex-to-real conversion is accomplished by a multiplexer that alternately selects between the real and imaginary values to form a real data stream consisting of real, unquantized data symbols.

The real sequence is then fed to the Viterbi algorithm via a selective bit inverter. The selective bit inverter corrects for an artifact of a differentially-encoded MSK signal: the differential decoding process combined with MSK-type modulation selectively inverts every other pair (in-phase and quadrature) of data symbols at the receiver output. The selective bit inverter prior to the Viterbi algorithm repeats this process yielding the original (non-inverted) data. The selectively-inverted real MF outputs are then sent to the VA section of the MLSE which attempts to equalize the channel. The hard-decision binary (0,1) data outputs of the MLSE are then converted from logical to arithmetic values, i.e., the symbols 0 and 1 are converted to $+1$ and -1, respectively.

To generate an estimate of the ISI, the hard-decision, real data is now converted into a differentially encoded complex baseband GMSK format signal. Thus, the MLSE output is effectively "remodulated." This is accomplished by again using a selective bit inverter to re-create the effect of the differential decoding and by using a real-to-complex converter which alternately places the data into real and imaginary components. The remodulated data is passed through a complex filter whose purpose is to synthesize the ISI present in the received signal at the MF output. The complex filter coefficients are a function of the estimated multipath channel impulse response and are found from channel soundings. Hence, the remodulated complex sequence is fed into a 9-tap complex FIR



MLSE
s_k coefficients from channel sounding
Demodulator output
Matched Filter
VA
ISI Generation
-
to de-interleaver and soft-decision decoder
Delay

Figure 3: Obtaining soft-decision information from an MLSE equalizer using ISI cancellation.

filter with an output w_n at time n given by

$$w_n = \sum_{k=-4}^{4} h_k a_{n-k}, \tag{5}$$

where a_n is the remodulated complex data signal and h_k is the kth complex FIR filter coefficient given by

$$h_k = \begin{cases} s_k & k > 0 \\ 0 & k = 0 \\ s_{-k} & k < 0 \end{cases} \tag{6}$$

where $s_k = h^*(-t) * h(t)|_{t=kT}$ [3].

The output of the 9-tap complex FIR filter is then converted into a real value in a manner identical to that of the complex-to-real conversion employed in the MLSE. As noted previously, this step is necessary since ultimately only real signals can be used as inputs to the convolutional decoder. The synthesized real ISI is then subtracted from the real MF output of the MLSE. This latter signal is suitably delayed to properly account for the processing delay in the VA section of the MLSE. The real reconstructed signal is then subtracted from the delayed real matched filter output signal. This operation removes the ISI components from the matched filter output. Because the result of this subtraction process is still selectively inverted due to the combination of the differential decoding process and the MSK modulation process, a final bit inversion selectively inverts the soft-decision data being fed to the decoder. The resulting real sequence is then fed to the soft-decision decoder. In the absence of ISI, the decoder input is simply the MF output. In the presence of ISI, the convolutional decoder input is theoretically the MF output with all the ISI terms removed.

2.3 Soft-Output Viterbi Algorithm (SOVA)

The Soft-Output Viterbi Algorithm can also be used for supplying soft information to the decoder [6]. Figure 4 illustrates an example of this algorithm. Essentially, the algorithm determines the metric difference for the paths merging into each state in the trellis. The

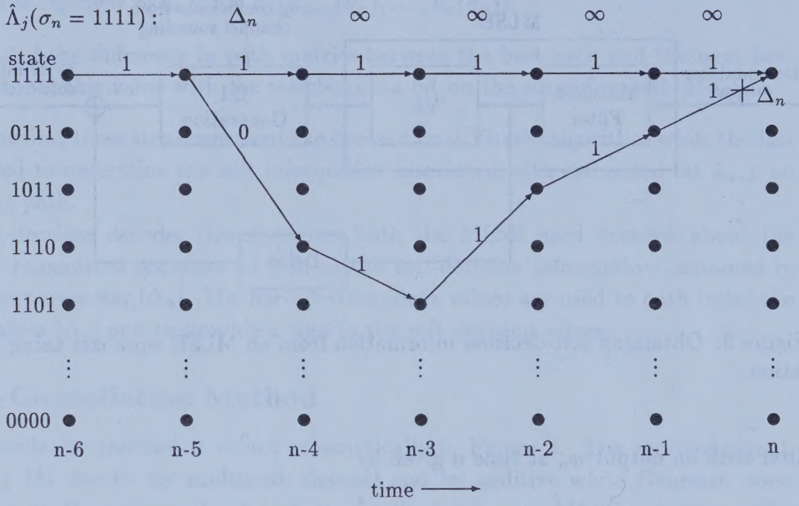


Figure 4: Example of the Soft-Output Viterbi Algorithm (SOVA).

merging paths at a given state are then compared to identify the information bits for which the paths differ. If an information bit along the surviving path differs from the information bit along the rejected path, the log-likelihood ratio for the surviving bit at time j, $\hat{\Lambda}_j$, is updated to reflect the path metric difference. The updating is performed so that only the worst log-likelihood ratio for each information bit in the surviving path for each state is retained. The likelihood ratios are updated until the paths merge or until the path memory truncation length τ is reached. That is, the soft-output Viterbi algorithm performs the following for each state σ_n at time n:

1. Compute the path metrics for the paths leading to state σ_n.
2. Compare the path metrics, select the best path and store the metric.
3. Store the survivor path $\hat{u}(\sigma_n) = (\hat{u}_{n-\tau}(\sigma_n), \ldots, \hat{u}_n(\sigma_n))$.
4. Find the difference in path metrics Δ_n between the best path and the next best path.
5. Compare the two paths merging into σ_n and update the likelihood ratio for each information bit on the surviving path for state σ_n if the information bits differ along the two paths. That is, for $j = n - L$ to $j = n - \tau$, if $\hat{u}_j^{(1)} \neq \hat{u}_j^{(2)}$, then update $\hat{\Lambda}_j(\sigma_n) = \min(\hat{\Lambda}_j(\sigma_n), \Delta_n/\alpha)$ where α is a constant used to prevent numerical overflow.

The SOVA, in practice, is similar to the metric difference method. In both cases, the metric difference between paths entering a state is used to derive soft information. For the SOVA, the metric difference between paths entering state σ_n is used to update soft information of *all* the estimated information bits along the surviving path which differ from the merging path. In the metric difference method, the metric difference between paths entering state σ_n at time n is used to calculate only the soft information associated with the estimated information bit at time $n-L$. The two algorithms require the same storage, 2^L registers to store the path decisions and 2^L registers to store the likelihoods. However, the SOVA requires more computations to determine the differing information bits.

3 Performance

Computer simulations were run to determine the performance of each of the three soft-output MLSE methods described above over simulated mobile radio channels. The channel model employed in the simulations was an L-ray multipath fading channel model consisting of a T-spaced tapped delay-line having L taps. Each of the tap coefficients are independent complex Gaussian random processes with a power spectral density given by

$$S(f) = \frac{3}{2\pi f_d}\left[1-\left(\frac{f}{f_d}\right)^2\right]^{-1/2}, \qquad -f_d < f < f_d; \tag{7}$$

f_d is the maximum Doppler shift (in Hz) given by

$$f_d = v/\lambda, \tag{8}$$

where v is the vehicle speed and λ is the wavelength of the carrier frequency; and T is the bit period. In the simulations, we have assumed that $f_dT=0.00018$. All synchronization was assumed to be ideal and training of the equalizer was performed in the absence of noise. The channel equalizer employed a 16-state MLSE algorithm and the convolutional code was the rate 1/2, $K=5$ convolutional code specified by the generator polynomials

$$\begin{aligned} g_o(D) &= 1 + D^3 + D^4 \\ g_1(D) &= 1 + D + D^2 + D^4. \end{aligned}$$

Figures 5-7 illustrate the performance of each of the soft-decision methods for $L=1$ (flat fading), $L = 2$, and $L = 3$, respectively. Shown in each of these figures is the channel bit error rate (BER) following the MLSE equalizer (no coding), the hard-decision decoded BER, and the soft-decision decoded BER for the metric difference approach, the ISI cancellation approach, and the Soft-Output Viterbi Algorithm (SOVA) approach. Observe the diversity advantage exhibited by the system as L is increased from 1 to 3. In addition, as the number of paths are increased, the performance differences between the three approaches become more pronounced, with the SOVA approach yielding the greatest improvement ($\approx$ 2 dB at 10^{-2} BER) for $L=3$.

For the shortest error event, the metric difference method and the ISI-cancellation method have been shown to yield the same soft information assuming that the channel

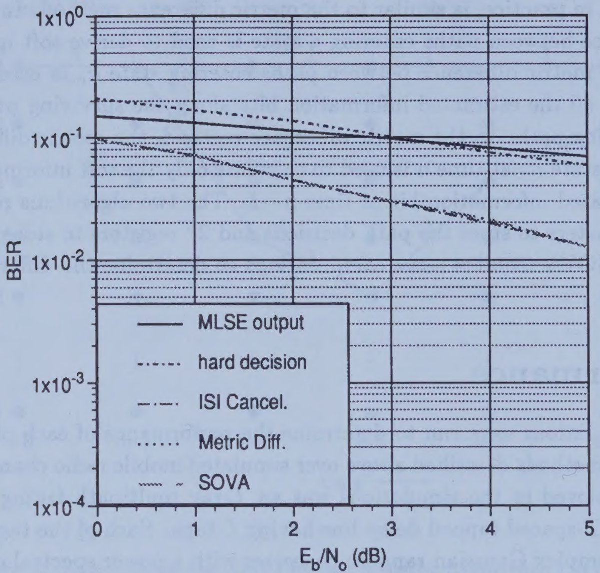


Figure 5. The bit error rate versus $E_b/\mathcal{N}_0$ for flat fading.

soundings are reliable [7]. Hence, the methods should have the same asymptotic performance. However, the simulations indicate that the metric difference method yields equivalent or superior performance under all channel conditions. Note that regardless of the method, a significant gain can be obtained for a coded system by using a soft-output MLSE rather than the conventional hard-output MLSE.

4 Conclusions

Soft information from the equalizer can be obtained using several methods which differ in how they extract the soft information as well as the reliability of the soft information. In the metric difference method, soft information about a given MLSE output symbol is extracted from the path metric difference at each state along the surviving path in the MLSE. In the ISI-cancellation method, the ISI is estimated and subtracted from the received symbol sequence, leaving a decoder input sequence with soft information. In terms of error performance, these methods compare favorably with the Soft-Output Viterbi Algorithm (SOVA) which is also based on the metric difference at each state along the surviving MLSE path, but generates the soft information by applying the metric difference to all dissimilar data symbols along the compared paths.

The complexity of the metric-difference and ISI-cancellation methods are both less than that of the SOVA, since the SOVA must determine the differing data bits along each pair of paths while these methods do not. In comparison to the hard-decision 16-state MLSE equalizer, the soft-decision MLSE equalizer using the metric difference method

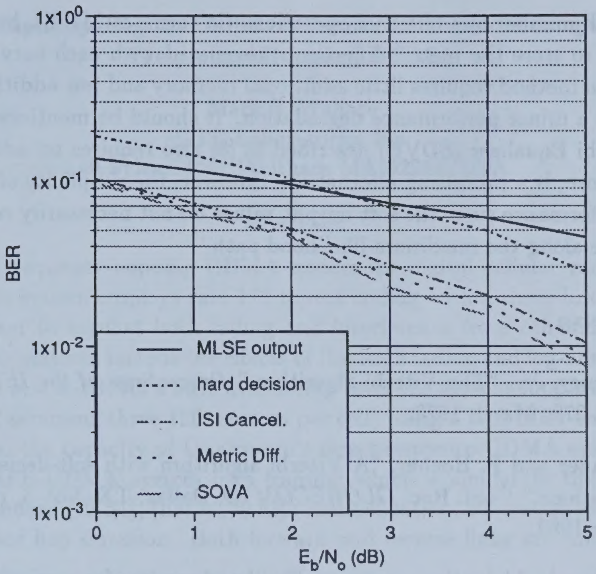


Figure 6. The bit error rate versus $E_b/\mathcal{N}_0$ for the $L=2$ channel.

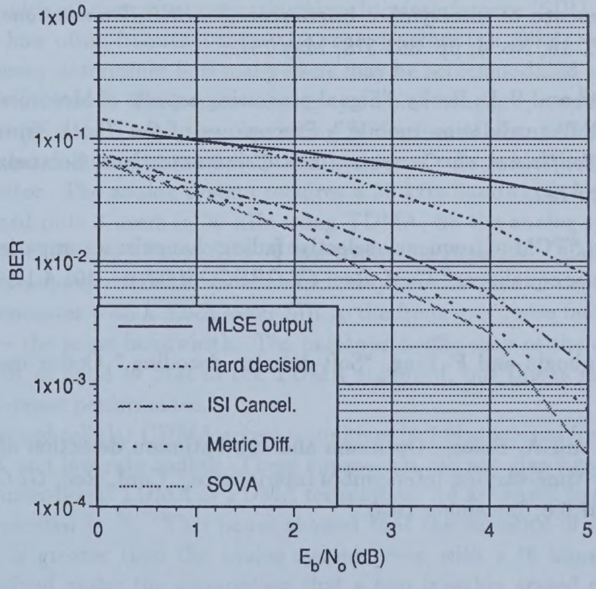


Figure 7. The bit error rate versus $E_b/\mathcal{N}_0$ for the $L=3$ channel.

requires a small number of additional operations (in terms of MIPS), but requires much more memory to store the metric differences associated with each surviving path. The ISI-cancellation method requires little additional memory and few additional operations, but the cost is a minor performance degradation. It should be mentioned that the Soft-Decision Viterbi Equalizer (SDVE) described in [8] also requires no additional memory and, in this sense, is a promising alternative. However, the simplicity of the SDVE is at the cost of performance since the soft output values do not necessarily correspond to the decisions made along the maximum-likelihood path.

References

[1] G. D. Forney, Jr., "The Viterbi algorithm," *Proceedings of the IEEE*, Vol. 61, No. 3, pp. 268-278, March 1973.

[2] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," Conf. Rec. *GLOBECOM'89*, Dallas, TX, Vol. 3, pp. 47.1.1-47.1.7, November 1989.

[3] G. Ungerboeck, "Adaptive maximum-likelihood receiver for carrier-modulated data-transmission systems," *IEEE Transactions on Communications*, **COM**-22, pp. 624-636, May 1974.

[4] G. D. Forney, Jr., "Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference," *IEEE Transactions on Information Theory*, **IT**-18, No. 4, pp. 363-378, May 1972.

[5] D. E. Borth and P. D. Rasky, "Signal processing aspects of Motorola's Pan-European digital cellular validation mobile," *Proceedings of the Tenth Annual International Phoenix Conference on Computers and Communications*, Scottsdale, AZ, pp. 416-423, March 1991.

[6] P. Hoeher, "TCM of frequency-selective fading channels: a comparison of soft-output probabilistic equalizers," Conf. Rec. *GLOBECOM'90*, pp. 401.4.1-401.4.6, December 1990.

[7] M. V. Eyuboglu and F. Ling, "Soft-decision decoding," Codex memo, January 11, 1990.

[8] W. Koch and A. Baier, "Optimum and sub-optimum detection of coded data disturbed by time-varying intersymbol interference," Conf. Rec. *GLOBECOM'90*, pp. 807.5.1-807.5.6, December 1990.

High-Capacity Digital Cellular Communications through Slow Frequency-Hopping CDMA

Mark S. Wallace
GTE Laboratories Inc.
40 Sylvan Rd., Waltham MA 02254, USA

Abstract

A slow frequency-hopping CDMA system for digital cellular radio is presented. The system employs rate 1/3 repeat coding with a three-branch adaptive equalizer to combat both fading and interference from other users. The capacity calculations include the effects of Rayleigh fading and log-normal shadowing with $\sigma = 8$ dB. At a 99% availability level the system supports 27 users per sector (assuming three 120° sectors per cell) using a 1.25 MHz band, which is similar to the capacity of Qualcomm's direct-sequence CDMA system. The proposed SFH-CDMA system uses framing which is similar to the proposed TDMA standard (IS-54), that is, 30 kHz channelization, 20 msec speech frames, and 6.7 msec hop duration. Both forward and reverse links are considered in the analysis.

1 Introduction and Summary

The fundamental tradeoff in selection of a multiple access technique for cellular radio is that of bandwidth efficiency vs. operating signal-to-interference (SIR). The operating SIR determines how often frequencies may be reused across the cellular system and the bandwidth efficiency determines how many users may be accommodated in a given bandwidth without interference. The current analog system and the proposed next generation TDMA standard (IS-54) [1] both have operating SIR values of about 17 dB which result in a frequency reuse factor of about 1/7 for 120° sectored cells, which is a reuse factor of 1/21 for each sector. The analog system requires a 30 kHz bandwidth for each user and the IS-54 standard puts 3 users in 30 kHz using TDMA. So the analog system capacity is 1 user per sector per 630 kHz, and IS-54 has triple this capacity. Qualcomm [2,3] has proposed a direct-sequence CDMA system which may provide greater capacity. This DS-CDMA system operates with a much lower SIR so the frequency reuse factor is 1, that is, every sector uses the same bandwidth. The bandwidth efficiency of the direct-sequence modulation is not as good as that of the TDMA standard, but this is more than offset by its frequency-reuse performance.

The first proposed cellular CDMA access techniques [4,5,6] employed frequency-hopping with m-ary FSK and low-rate coding. These approaches did not give significantly better capacity than conventional TDMA or FDMA techniques. An approach to slow frequency-hopping was presented in [8]. This paper showed that the capacity of a random slow-hopping system is greater than the analog system, even with a 16 kbps vocoder. The capacity was derived under the assumption that a hop is either erased or received correctly.

This paper presents a CDMA system which uses slow-frequency-hopping with coherent QPSK modulation and soft decision decoding. The hopping is slow so synchronization of the hops is not difficult and users within a sector can use orthogonal hopping patterns on both the forward (base–mobile) and reverse (mobile–base) links. As with the Qualcomm DS-CDMA system, every frequency is reused in every sector of every cell. In order to operate with high levels of interference from other users, each block of symbols is transmitted consecutively on three different frequencies. The blocks are sampled and buffered by the receiver, and are then equalized in parallel using a three-branch equalizer. This approach provides diversity to overcome the Rayleigh fading effects as well as redundancy to mitigate the interference effects.

2 Mobile Channel Characteristics

The mobile channel is characterized by Rayleigh fading, log-normal shadowing and inverse k-th power distance dependence of the median path loss. If we neglect the dispersion in the channel then the received signal level for constant transmit power is given by

$$P_s(t) = \alpha |X(t)|^2 Z(t) d^{-k} \tag{1}$$

where α is a constant, $X()$ is a complex Gaussian unit-variance process (this results in Rayleigh fading), $Z()$ is a log-normal process, and d is the link distance. The Rayleigh fading is generally much faster than the log-normal shadowing process. The distance d is a function of time, but is much slower than the log-normal process and may be considered constant. We assume throughout that the log-normal shadowing process has $\sigma = 8$ dB and that the propagation exponent k is four.

The Doppler spectrum of the Rayleigh fading process is derived assuming uniform scattering from all angles [7] and the channel delay-power profile (i.e., time dispersion) has a Gaussian shape. The typical mobile channel delay-power profile does not have this shape, but most channels with about the same RMS delay spread produce roughly equivalent channel-to-channel correlation functions. The analysis assumes a channel 2σ delay spread of 1 μsec and a maximum Doppler frequency of 20 Hz. The system performance is worst under slow fading conditions with small dispersion since little diversity may be available over the system bandwidth. These values are taken to represent a reasonable worst case. The fading may be slower than 20 Hz, but since the power control process operates at the frame rate (50 Hz), slow fading may be compensated somewhat by the power control.

The Rayleigh fading process is assumed to be constant over each transmitted slot. The fading experienced by different transmissions is correlated, and the correlation is a function of the time between the transmissions and the frequency separation of the channels. The correlation between the complex Gaussian amplitudes on two transmissions is given by

$$\rho(\Delta t, \Delta f) = J_0(2\pi f_d \Delta t) e^{-(2\pi \Delta f \tau)^2/2} \tag{2}$$

where Δt is the time between the transmissions, Δf is the frequency separation, τ is the RMS delay spread, J_0 is the zero-th order Bessel function of the first kind, and f_d is the maximum Doppler frequency which is derived from the vehicle velocity. This equation holds under the assumption that the channel impulse response is the sum of a large number of impulses at very closely spaced delays (i.e., delays much closer than the reciprocal of the system bandwidth), where the impulse amplitudes are independent complex Gaussian processes, each with the same fading statistics and average power given by the delay-power profile.

3 System Description

The system employs synchronous slow frequency-hopping. All channels are used in every sector of every cell. Within each sector the hopping patterns used are orthogonal, i.e., there are no collisions among transmissions within the same sector. So with N_c channels available the i-th user in the j-th sector uses a hopping pattern

$$f_{ji}(t) = (f_j(t) + i) \bmod N_c \tag{3}$$

where t is the slot index. The random hopping pattern $f_j()$ is a derived from a long shift-register sequence, such that the hopping patterns for each sector in the system are effectively independent. No special properties are required for these sequences, that is, there is no attempt to select special sequences which have cross-correlation statistics which are "better" in some sense than those of independent sequences. (In fact, the sequences employed in the simulation do not even preclude consecutive transmissions on the same channel.) However, we note that only one independent sequence is required per sector, and the system can reuse sequences in the same manner as frequencies are reused in the analog cellular system. This means that only 25 or so different sequences are required system-wide and so it may be possible to select a set of sequences with better correlation properties. Each speaker is assumed to be active 50% of the time. It is assumed that no transmissions are made if the speaker is inactive.

3.1 Handoffs and power control

Each mobile user measures the path loss to all nearby base stations, averaged over the Rayleigh fading process. The user requests a channel from the base station with the minimum path loss. It is assumed in the analysis here that handoffs are perfect and instantaneous, so the path loss (due to distance and shadowing) between a given user and its base station is no greater than the path loss between that user and any other base station. In an actual system soft handoffs [2] would be employed, that is, a call will be up on two different base stations simultaneously when the user is on the boundary between the two cells. Soft handoffs can improve capacity since the additional traffic may be offset by reduced SNR requirements, but they are not considered in the analysis here.

Adjustment of the transmitted power levels on both the forward and reverse links is essential to reduce the mutual interference between users in different sectors. The power control is assumed to operate on the components of path loss due to distance and the log-normal shadowing process. On the reverse link the power control algorithm adjusts the mobile transmit power such that the average power received by the base station is constant. On the forward link the derivation of a good power control algorithm is not as straightforward. The general goal is to decrease the power transmitted to users which are close to the base and see little interference from other bases. The power transmitted to a user even very close to a base station cannot be decreased too far without encountering interference from the other sectors of the same cell (the isolation between sectors is assumed to be 20 dB). The power control algorithm used in the analysis here balances the received power from the home base and the next most powerful base, but limits the power control range to 10 dB. That is, the power adjustment from the nominal transmit level is

$$P_{cf} = \max\left[0.1, \frac{Z_1(t)d_1^{-k}}{Z_0(t)d_0^{-k}}\right] \tag{4}$$

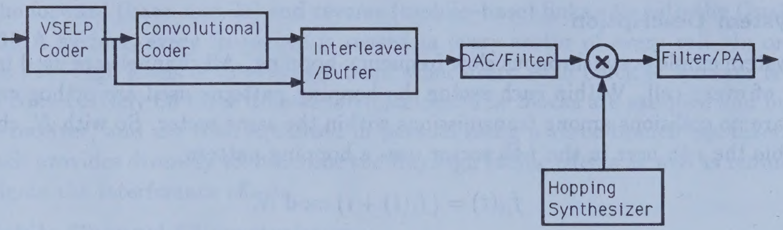


Figure 1: Transmitter structure.

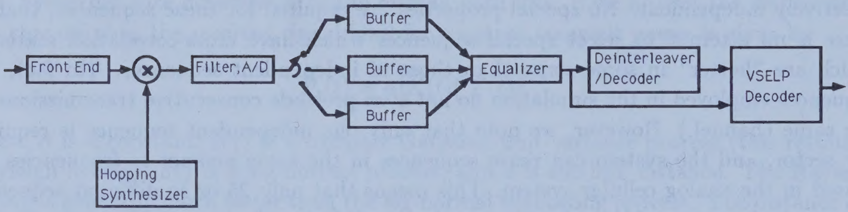


Figure 2: Receiver structure.

where the zero subscript indicates the values of distance and log-normal shadowing for the home base and the one subscript indicates values for the base with the next smallest path loss. Since frequency-hopping systems generally perform best if the interference is concentrated into a single hop, it is not clear that the approach of balancing power is best, but it is certainly better than no power control.

3.2 Signal processing

The transmit signal processing is illustrated in Figure 1. The first stage is a 7950 kbps speech coder (e.g., VSELP) which produces 159 bits for each 20 msec speech frame. The most significant half of these bits (denoted Class 1) are encoded with a convolutional encoder (rate 1/2, k=6). A parity check word is added to the class 1 bits before convolutional encoding. These bits are then interleaved over two 20 msec frames. The data bits are then combined with roughly 20% reference plus some other overhead and modulated using coherent QPSK. Each frame of modulated data consists of $N_s = 180$ symbols. The interleaved frame is transmitted three times at three different frequencies, so the hop duration is 6.7 msec. (We use the term frame to denote a 20 msec interval, and hop for a 6.7 msec interval.) The details of the modulated frame structure and reference are not used in the analysis. We simply assume that groups of 3 hops may be equalized with an implementation loss of 2 dB relative to the theoretical performance of maximal-ratio combining. We further assume that the interference may be modelled as Gaussian noise of the same power. Some care must be taken to avoid situations where the same reference symbols are employed by two users who may interfere with one another, since the equalizer may not be able to estimate the fading channel properly.

At the receiver (Figure 2), the received signal is dehopped and groups of three hops are sampled, buffered, and then equalized simultaneously. The equalizer outputs are then deinterleaved and the Class 1 bits are decoded.

4 Signal and interference power calculation

The received signal power is assumed to be constant over each hop. The correlation between the Rayleigh fading on different hops is a function of the time between the hops and the hop frequencies. So for the i-th hop the signal power is

$$P_s(i) = |X_i|^2 Z_0 d_0^{-k} P_c P_{ce}(m) \tag{5}$$

where X_i is complex Gaussian ($\mu = 0$, $\sigma = 1$) and

$$\mathrm{E}[X_i X_j] = \rho(|i-j|T_h, |f_i - f_j|) \tag{6}$$

where $\rho()$ is given by (2), T_h is the hop duration, and f_i is the channel for the i-th hop. The other terms in (5) are the log-normal shadowing Z_0, the link distance d_0, the power control P_c, the frame index $m = \lfloor i/3 \rfloor$, and the power control error $P_{ce}(m)$ which is log-normal ($\sigma = 1$). Since the power control is adjusted each frame it is reasonable to assume that the error $P_{ce}(m)$ is independent for different m.

On the forward link (5) reduces to

$$P_s(i) = |X_i|^2 Z_1 d_1^{-k} P_{ce}(m) \tag{7}$$

using the forward link power control definition (4), where Z_1 and d_1 are the distance and log-normal shadowing from the user to the next best base, that is, the base with the smallest path loss other than the home base. On the reverse link the received power is

$$P_s(i) = |X_i|^2 P_{ce}(m) \tag{8}$$

since the power control eliminates the effects of distance and log-normal shadowing.

Next we calculate the interference power seen by a user in sector 0. For convenience we refer to the user of interest simply as the user (and use the index 0 for this user in equations) and all other users of the system as interferers. On the forward link (base–mobile) the interference power on the i-th hop is given by

$$P_n(i) = \sum_{s>0} |X_{is}|^2 Z_{s0} d_{s0}^{-k} \Gamma(s, i, f_0(i))\, \lambda(u) P_{cf}(s,u) P_{ce}(s,u,i) G_s(0) \tag{9}$$

where the summation is over all sectors s other than the sector of the user of interest,

X_{is} = amplitude of the Rayleigh fading between the s-th sector and the user for the i-th hop
Z_{s0} = log-normal shadowing value for the path between the user and sector s
d_{s0} = distance between the user and the antenna of sector s
$\Gamma(s, i, f_0(i))$ = binomial r.v. which specifies whether an interferer is present in sector s at time i in channel $f_0(i)$
u = index of the interferer
$\lambda(u)$ = binomial r.v. for the voice activity factor of interferer u ($\mathrm{P}[\lambda(u) = 1] = \mathrm{P}[\lambda(u) = 0] = 0.5$)
$P_{cf}(s,u)$ = forward-link power control for the interferer
$P_{ce}(s,u,i)$ = error in the power control (assumed log-normal with $\sigma =$ 1 dB)
$G_s(0)$ = gain of the sector s antenna seen by the user.

Since the hopping patterns of each sector s are independent, the r.v. $\Gamma(s, i, f_0(i))$ is 1 with probability N_u/N_c where N_c is the number of channels available and N_u is the number of users per sector. $\Gamma()$ is independent from hop-to-hop since the hopping pattern of sector s is independent from that of sector 0.

The Rayleigh fading process X_{is} is correlated for different hops in the same way that the signal is correlated (6), even though the transmissions on different hops are generally for different interferers. Note that $\lambda()$, Z_{s0}, $P_c(s, u)$, and d_{s0} are slowly-varying parameters which are fixed in the outer loop of the simulation. The system availability is determined by the fraction of these parameters which result in acceptable BER statistics.

On the reverse link (mobile–base) the interference power is

$$P_n(i) = \sum_{s>0} |X_{i0}(u)|^2 Z_{0u} d_{0u}^{-k} \Gamma\left(s, i, f_0(i)\right) \lambda(u) P_{cr}(s, u) P_{ce}(s, u, i) G_0(u) \tag{10}$$

where

u = index of the interferer in sector s for the i-th hop

$X_{i0}(u)$ = value of the Rayleigh fading between the 0-th sector and interferer u

Z_{0u} = log-normal shadowing value for the path between the interferer u and sector 0

d_{0u} = distance between the interferer and the antenna of sector 0

$P_{cr}(s, u)$ = reverse-link power control for the interferer

$G_0(u)$ = gain of the sector 0 antenna seen by the interferer.

The other parameters in (10) are as defined in the forward link equation.

4.1 Equalizer outputs and soft decision scaling

The received samples for the i-th hop are modeled as

$$r_i(j) = a_{mj} + \frac{\sqrt{P_n(i)}}{\sqrt{L P_s(i)}} n_j \tag{11}$$

where $m = \lfloor i/3 \rfloor$ is the frame index, a_{mj} is the QPSK symbol value, the n_j are independent complex Gaussian r.v.'s ($\mu = 0, \sigma = 1$), and $L = 0.63$ is the implementation loss (2 dB). Note that this equation assumes that the phase of the received symbols is recovered perfectly, that the sampling phase is perfect, and that no ISI is present, either channel-induced or due to filtering. Also note that on a given hop the interference and fading effects are approximated using additive Gaussian noise with average power given by the interference-to-signal ratio. If a single interferer dominates, then the interference amplitude statistics may be nearly constant envelope rather than Gaussian. However, since the equalizer combines interference from three hops the approximation is very good even in this case. (For a Rayleigh fading signal, even the performance with a single Rayleigh-fading QPSK interferer is very close to the performance in Rayleigh fading noise. They differ only by a fraction of a dB for ABER less than 1% or so.)

The output of the equalizer for the m-th frame is $v_m(j); j = 1, \ldots, N_s$, where $v_m(j)$ is

$$v_m(j) = \sum_{i=3m}^{3m+2} \frac{P_s(i)}{P_n(i)} r_i(j). \tag{12}$$

The $v_m()$ include scaling by the post-detection SNR to generate the proper soft decisions for maximum-likelihood decoding. The inputs to the Viterbi decoder are the equalizer outputs deinterleaved over two frames.

5 Simulation description and system performance measure

The simulation assumes hexagonal cells with 120 degree sectors. The isolation between sectors is assumed to be 20 dB and the antenna patterns are assumed to be constant for all angles within the 120 degree sector, so the antenna gain factor $G_i()$ in equations (9) and (10) is 1 when the user is within the antenna beamwidth and 0.01 otherwise. The simulation considers a total of 19 cells or 57 sectors, the cell in which the user is located plus 2 tiers of interfering cells. A fixed number of users are placed randomly within each sector. These users are then assigned to sector antennas based on minimum path loss including the log-normal shadowing effects.

The simulation divides the random parameters into three levels based on their relative rates. The slowest parameters are the locations of the mobile units, the values of log-normal shadowing between each mobile and base, and the voice activity, that is, whether a given mobile is talking or listening (a 50% duty factor is assumed). These parameters are fixed in the outer loop of the simulation. The next level of random parameters consists of the hopping patterns, the Rayleigh fading of both the signal and interferers, and the power control error process. The error rates of the coded and uncoded bits are determined by averaging over these parameters. In order to compute the coded bit error rates, the user hopping pattern for 6 consecutive hops is selected randomly. Then a set of 6 correlated Rayleigh fading values are selected for each sector, and the average signal and interference power are computed. Finally, the inputs to the Viterbi decoder are modelled by a scaled bit value plus Gaussian noise, where the SNR is computed from the correlated Rayleigh fading on the various hops and assumes maximal-ratio combining is performed by the equalizer. For each value of hopping patterns and Rayleigh fading, a block of 88 bits is decoded.

The speech quality is deemed acceptable or unacceptable based on the error rates averaged over the Rayleigh fading and hopping patterns (as well as the bit-to-bit Gaussian noise). Specifically, if the average uncoded (class 2) BER is about 3% or less and no more than 1% of the speech frames have errors in the class 1 bits after convolutional decoding, then the speech quality is acceptable. The system performance is evaluated based on availability, which is defined as the fraction of the vehicle location/shadowing cases which result in acceptable speech quality. The capacity of the system is determined by the number of users which are allowable to achieve an availability of say 90% or 99%. The outer loop of the simulation is repeated 1000 times, i.e., 1000 different cases of locations/shadowing are examined. The next loop is repeated 500 times (hopping patterns/Rayleigh fading).

6 Results and Conclusions

In Figure 3 the distribution of signal-to-interference (SIR) for a single hop is given. These curves show that it is reasonably likely that a hop sees a poor SIR, e.g., on the forward link at a 99% availability a hop sees an SIR worse than 0 dB slightly less than half of the time. The curves are helpful to get a feel for the conditions under which the system is operating, but the system performance is not characterized by these curves since the correlation between the SIR's on different hops is important for both the equalizer and

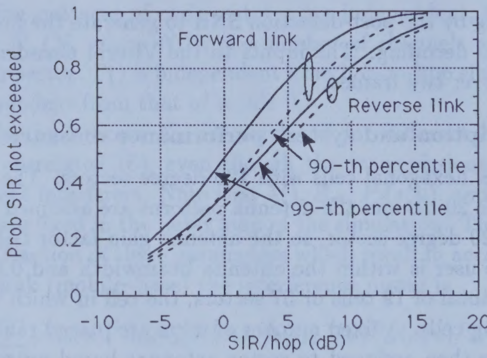


Figure 3: Cumulative distribution of per hop signal-to-interference ratio (SIR) with 30 users/sector. Results given for the 90-th and 99-th percentiles of the log-normal/location parameters, and for both forward and reverse links

	Capacity (users/sector)		Class 2 ABER	
Availability	Forward	Reverse	Forward	Reverse
90%	40	40	2.6%	3.0%
99%	27	35	2.5%	3.0%

Table 1: Capacity in users/sector for forward and reverse links under 90 and 99% availability requirements. Availability defined as fraction of cases with no greater than 1% frame deletion rate. Class 2 (uncoded) ABER included for reference.

the decoder performance. The difference between the 90-th and 99-th percentiles is much greater for the forward link than for the reverse link. This effect is observed because the interference level at a given mobile unit is greatly affected by the log-normal shadowing loss to a few nearby base stations. In contrast, on the reverse link, the performance is affected by the log-normal shadowing for a large number of users and a single base. Poor performance is experienced only if a significant fraction of the nearby interferers happen to have small shadowing loss to the base.

The frame deletion rate for different number of users per sector is given in Figure 4. Error detection is employed on the most significant half of the speech bits (the Class 1 bits), and the entire speech frame is deleted (i.e., the previous frame is repeated) if any errors are detected in these bits after convolutional decoding.

The capacity results for both forward and reverse links are given in Table 1. Note that the maximum capacity is 40 users/sector, since each user is assumed to have a hopping pattern which is orthogonal to all other users in the same sector and there are only 40 available channels.

Table 2 gives an approximate comparison of the capacity of the proposed system with the IS-54 and DS-CDMA systems. Accurate comparison of the various access technologies

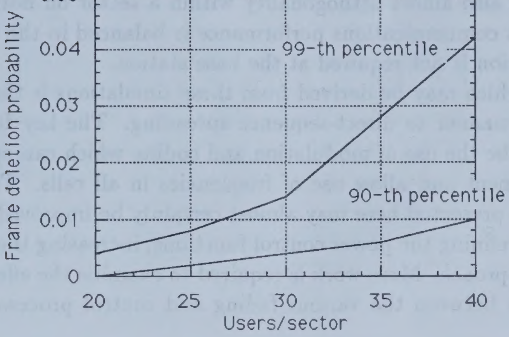


Figure 4: Probability of frame deletion (i.e., fraction of speech frames with uncorrected Class 1 bit errors) as a function of the number of users per sector for the 90-th and 99-th percentiles of the log-normal/location parameters on the forward link.

System	Availability	
	90%	99%
SFH-CDMA	40	27
IS-54	6	—
Qualcomm DS-CDMA	40*	

Table 2: Capacity comparison — number of users/sector/1.25MHz. (* - The availability of the DS-CDMA system at capacity has not been stated.)

is difficult because of the large number of parameters for both the systems and the environment. The design goal for the IS-54 system is to provide acceptable voice quality at an average SIR of 17 dB (averaged over the Rayleigh fading). Under the propagation assumptions we have made the SIR is better than 17 dB about 90% of the time with a sector frequency reuse factor of 1/21, so if this goal is achieved then the IS-54 system provides the capacity given in Table 2 at an availability level of 90%. It does not appear likely that the IS-54 system will provide reasonable capacity at 99% availability. The analysis which gives the DS-CDMA capacity numbers has not been laid out in great detail (nor has the system design) so it is difficult to determine what availability is achieved.

There are some advantages of a slow-frequency-hopping CDMA system over a DS-CDMA system. The bandwidth allocated to an SFH-CDMA system need not be contiguous. This facilitates coexisting with other services and systems and improves the efficiency of spectrum use. In addition the SFH system is more robust to interference and power control failures. If a DS-CDMA user transmits at high power near a base station, the entire system may experience an outage. In the SFH system, the system can operate error-free even if a single user completely blocks one of the channels. The

SFH-CDMA system also allows orthogonality within a sector on both the forward and reverse links, so that communications performance is balanced in the two directions and dual diversity reception is not required at the base station.

One conclusion which may be derived from these simulations is that high capacity is not linked in some manner to direct-sequence spreading. The key factor in improving capacity appears to be the use of modulation and coding which can operate at low SNR in a fading environment and allow use of frequencies in all cells. The capacity of the SFH-CDMA system presented here may almost certainly be improved by using different hopping sequences, refining the power control functions, increasing the hop rate, or using a different coding approach. More work is required to examine the effects of equalization and the interactions between the various fading and control processes for a variety of channel conditions.

References

[1] "Dual-mode mobile station — base station compatibility standard," TIA IS-54 Rev A (TIA PN 2398).

[2] "An Overview of the Application of Code Division Multiple Access to Digital Cellular Systems and Personal Communications Networks," submission to the Cellular Telecommunications Industry Association (CTIA) "CDMA Technology Investigation Committee," Qualcomm, Inc., 10555 Sorrento Valley Road, San Diego CA 92121, January 11, 1991.

[3] K. S. Gilhousen, I. M. Jacobs, R. Padovani, A. J. Viterbi, L. A. Weaver, and C. E. Wheatley, "On the Capacity of a Cellular CDMA System," *IEEE Trans. Veh. Technol.*, vol. VT-40, pp. 303–312, Jun. 1991.

[4] G. R. Cooper and R. W. Nettleton, "A spread-spectrum technique for high-capacity mobile communications," *IEEE Trans. Veh. Technol.*, vol VT-27, pp. 264–275, Nov. 1978.

[5] D. J. Goodman, P. S. Henry, and V. K. Prabhu, "Frequency-hopped multilevel FSK for mobile radio," *B. S. T. J.*, vol. 59, pp. 1257–1275, Sept. 1980.

[6] G. Einarsson, "Coding for a multiple-access frequency-hopping system," *IEEE Trans. Commun.*, vol. COM-32, no. 5, May 1984.

[7] W. C. Jakes Jr. *et al.*, *Microwave Mobile Communications*, New York: Wiley, 1974.

[8] D. Verhulst, M. Mouly, and J. Szpirglas,"Slow frequency hopping multiple access for digital cellular radiotelephone," *IEEE Journal on Selected Areas in Communications*, vol. SAC-2, no. 4, pp. 563–574, July 1984.

ON THE CPM FAMILY AND ITS PATHOLOGICAL MEMBERS

Quinn Li and Bixio Rimoldi
Electronic Systems and Signals Research Laboraory
Department of Electrical Engineering
Washington University
St. Louis, MO 63130
October 2, 1991

Abstract

Continuous Phase Modulation (CPM) is a set of bandwidth and energy efficient constant-envelope modulation schemes. CPM is reviewed here as an encoded modulation scheme, where the encoder is a convolutional encoder over the ring of integers modulo P and the modulator is memoryless. Some convolutional encoders (over fields) have the undesirable property of being catastrophic; we will see that some CPM schemes have an equivalent property. Special attention will be given to a class of CPM schemes denoted L-REC. We will determine which schemes among this class are catastrophic and derive a precoder that transforms all schemes of this class into equivalent non-catastrophic schemes.

1 Background

Constant-envelope digital modulation schemes such as phase-shift keying (PSK) and frequencey-shift keying (FSK) are good candidates for digital transmission over nonlinear and/or fading channels. By smoothing the information-carrying phase, the normalized bandwidth of a constant-envelope digital modulation scheme can be reduced. The resulting class of modulation schemes is continuous phase modulation (CPM). This is a very large class of constant-envelope modulation schemes containing members which are both bandwidth and energy efficient.

The general form of the transmitted signal can be described by

$$s(t, \boldsymbol{U}) = \sqrt{2E/T_s} \cos(2\pi f_o t + \psi(t, \boldsymbol{U}) + \psi_o),\ t \geq 0, \tag{1}$$

where

$$\psi(t, \boldsymbol{U}) = 4\pi h \sum_{i=0}^{\infty} U_i f(t - iT_s),\ t \geq 0 \tag{2}$$

is the information-carrying phase, E is the symbol energy, T_s is symbol time, f_o is the carrier frequency, and ψ_o is an initial phase shift which can be assumed to be zero without loss of essential generality. The parameter h is referred to as the *modulation index*, and is defined by

$$h = K/P$$

where K and P are relative positive prime integers. The data sequence is simply an M-ary sequence with $U_i \in \{0, 1, \ldots, M-1\}$, and the phase response $f(t)$ satisfies

$$f(t) = \begin{cases} 0, & t \leq 0 \\ \frac{1}{2}, & t > LT_s, \end{cases} \tag{3}$$

where L is a positive integer called the *memory* of the system.

Using (3) in equation (2), the information-carrying phase in the n-th interval can be described as

$$\begin{aligned} \psi(t, \boldsymbol{U}) &= 2\pi h \sum_{i=0}^{n-L} U_i \\ &+ 4\pi h \sum_{i=n-L+1}^{n} U_i f(t - iT_s),\ nT_s \leq t < nT_s + T_s. \end{aligned} \tag{4}$$

Changing the information-carrying phase by a multiple of 2π does not affect the transmitted signal. Therefore an equivalent description of the transmitted signal is

$$s(t, \boldsymbol{U}) = \sqrt{2E/T_s} \cos(2\pi f_o t + \psi'(t, \boldsymbol{U}) + \psi_o),\ t \geq 0, \tag{5}$$

where

$$\begin{aligned} \psi'(t, \boldsymbol{U}) &= 2\pi h R_P[\sum_{i=0}^{n-L} U_i] \\ &+ 4\pi h \sum_{i=n-L+1}^{n} U_i f(t - iT_s),\ nT_s \leq t < nT_s + T_s, \end{aligned} \tag{6}$$

and $R_P[.]$ denotes the the remainder after division of the enclosed number by P.

Thus the transmitted signal in the n-th interval depends only on the input via the $(L+1)$-tuple $(U_n, U_{n-1}, \ldots, U_{n-L+1}, R_P[\sum_{i=0}^{n-L} U_i])$. This shows that a general CPM scheme can be represented as in Figure 1.

Figure 1 suggests that CPM is itself an encoded modulation scheme. Here the purpose of the encoder is to reduce both the probability of error and the bandwidth of the transmitted signal. Some advantages of this viewpoint are discussed in [1].

Of particular interest is the case where $P = M$, since then the encoder is a convolutional encoder over the ring of integers modulo M. Moreover, if M is a prime number, then the encoder is the usual convolutional encoder. While requiring $P = M$ gives up some freedom, the class of CPM schemes is so large that it is an acceptable loss for the gain due to the additional algebraic structure.

It is well known that some convolutional encoders have the undesirable property of being catastrophic. A convolutional encoder is catastrophic if there exists two infinite input sequences, say $\boldsymbol{U}$ and $\boldsymbol{U}'$, that differ in an infinite number of positions, such that the corresponding output sequences differ only in a finite number of positions. Such encoders are undesirable since a finite number of channel errors can cause an infinite number of information symbol errors.

It is natural to ask whether or not there are members within the CPM family that have similar pathologies. We will see in the next section that the answer is yes.

We will start by defining what is meant by catastrophic CPM schemes. The issue here is that for CPM schemes the encoder comes with the associated modulator as shown in Fig. 1. Therefore we are interested in considering the cascade of the encoder and the modulator as opposed to the encoder alone. Then we will see that some CPM schemes are indeed catastrophic, and finally we will show how to precode a catastrophic CPM scheme in such as way that the new scheme, which is equivalent to the original one, is not catastrophic.

2 Catastrophic CPM schemes

Definition: A CPM system is said to be catastrophic if there exists a pair of input sequences, say $\boldsymbol{U}$ and $\boldsymbol{U}'$, which differ in an infinite number of places such that the corresponding transmitted signals $s(t,\boldsymbol{U}), s(t,\boldsymbol{U}')$ have finite Euclidean distance.

In order to determine whether or not a CPM scheme is catastrophic we need to be able to compute the Euclidean distance between all pairs of transmitted signals. If $\boldsymbol{U}$ and $\boldsymbol{U}'$ are two data sequences, the Euclidean distance between the corresponding transmitted signals is:

$$D^2(\boldsymbol{U},\boldsymbol{U}') = \sum_{i=0}^{\infty} D_i^2(\boldsymbol{U},\boldsymbol{U}'),$$

where

$$\begin{aligned} D_i^2(\boldsymbol{U},\boldsymbol{U}') &= \int_{nT_s}^{nT_s+T_s} [s(t,\boldsymbol{U}) - s(t,\boldsymbol{U}')]^2 dt \\ &= 2E[1 - \frac{1}{T_s}\int_{iT_s}^{iT_s+T_s} \cos(\psi'(t,\boldsymbol{U}) - \psi'(t,\boldsymbol{U}'))dt] \\ &= 2E[1 - \frac{1}{T_s}\int_{iT_s}^{iT_s+T_s} \cos(\Delta\psi'(t,\boldsymbol{\varepsilon}))dt]. \qquad (7) \end{aligned}$$

From equation (6)

$$\begin{aligned} \Delta\psi'(t,\boldsymbol{\varepsilon}) &= \psi'(t,\boldsymbol{U}) - \psi'(t,\boldsymbol{U}') \\ &= 2\pi h(R_P[\sum_{i=0}^{n-L} U_i] - R_P[\sum_{i=0}^{n-L} U_i']) \\ &+ 4\pi h \sum_{i=n-L+1}^{n} (U_i - U_i')f(t-iT_s) \qquad (8) \end{aligned}$$

Again, we can change $\Delta\psi'(t,\boldsymbol{\varepsilon})$ by a multiple of 2π without affecting the right hand side of (7). Thus, with a slight abuse of notation we write

$$\begin{aligned} \Delta\psi'(t,\boldsymbol{\varepsilon}) &= 2\pi h R_P[\sum_{i=0}^{n-L}(U_i - U_i')] \\ &+ 4\pi h \sum_{i=n-L+1}^{n} (U_i - U_i')f(t-iT_s) \\ &= 2\pi h R_P[\sum_{i=0}^{n-L} \varepsilon_i] \end{aligned}$$

$$+ \quad 4\pi h \sum_{i=n-L+1}^{n} \varepsilon_i f(t - iT_s),\ nT_s \leq t < nT_s + T_s. \tag{9}$$

$\boldsymbol{\varepsilon} = \boldsymbol{U} - \boldsymbol{U}'$ is the difference of the input sequences.

Proposition
The necessary and sufficient condition for a CPM scheme to be catastrophic is that there exists a sequence $\{\varepsilon_i\}_{i=0}^{\infty}$ and an integer N such that $\varepsilon_i \neq 0$ infinitely often and $\Delta\psi(t, \boldsymbol{\varepsilon}) = 2\pi c$ for some integer c and $t \geq NT_s$.

Proof:
The Euclidean distance between two signals within the interval $[nT_s, nT_s + T_s)$ is defined in equation (7). Now if there exists a sequence $\{\varepsilon_i\}_{i=0}^{\infty}$ with an infinite number of nonzero terms and an integer N such that $\Delta\psi(t, \boldsymbol{\varepsilon}) = 2\pi c$ for some integer c and $t \geq NT_s$, we are going to have zero Euclidean distance in the symbol intervals thereafter, and therefore the scheme is catastrophic. On the other hand, if a CPM system is catastrophic, then there exists two input sequences that differ in infinite number of places, but have zero distance for $t \geq NT_s$. From (7) we see that the distance in the i-th interval is zero iff the phase difference in the same interval is a multiple of 2π. □

For the rest of the paper we consider the subset of CPM schemes characterized by:

$$f(t) = \begin{cases} 0 & t \leq 0 \\ \frac{t}{2LT_s} & 0 \leq t < LT_s \\ \frac{1}{2} & t \geq LT_s. \end{cases} \tag{10}$$

These schemes are denoted L-REC since the impulse response is the integral of a rectangle of width L. Now equation (9) becomes

$$\Delta\psi(t, \boldsymbol{\varepsilon}) = 2\pi h R_p[\sum_{i=0}^{n-L} \varepsilon_i] + \frac{2\pi h}{L} \sum_{i=n-L+1}^{n} \varepsilon_i \frac{t - iT_s}{T_s}, nT_s \leq t < nT_s + T_s. \tag{11}$$

The necessary and sufficient condition for this scheme to be catastrophic is that there exists a sequence $\{\varepsilon_i\}_{i=0}^{\infty}$ with infinite nonzero symbols such that:

1. There is an N such that for all $n \geq N$, the phase difference in the n-th interval is constant, i.e.,
$$\sum_{i=n-L+1}^{n} \varepsilon_i = 0, n \geq N.$$
An equivalent condition is
$$\sum_{i=N-L+1}^{N} \varepsilon_i = 0, \tag{12}$$
and the sequence $\{\varepsilon_{N-L+i}\}_{i=1}^{\infty}$ is periodic with period L.

2. The phase difference is equal to $2\pi c$ at $t = NT_s$, i.e.
$$hR_P[\sum_{i=0}^{N-L} \varepsilon_i] + \frac{h}{L} \sum_{i=N-L+1}^{N} \varepsilon_i(N-i) = c \tag{13}$$

The second condition can be considered as consisting of two terms: The term due to the last L inputs, namely $\sum_{i=N-L+1}^{N} \varepsilon_i(N-i)$, and the term due to all previous inputs, namely $R_p[\sum_{i=0}^{N-L} \varepsilon_i]$.

Let $a = R_P[\sum_{i=0}^{N-L} \varepsilon_i]$, where $a \in \{0, 1, \ldots, P-1\}$, and $b = \sum_{i=N-L+1}^{N} \varepsilon_i(N-i)$. Equation (13) with $h = \frac{K}{P}$ can be written as:
$$\frac{K}{P}(\frac{aL+b}{L}) = c. \tag{14}$$
It is clear that we can force a to take on any desired value in $\{0, 1, \ldots, P-1\}$, provided that N is large enough, i.e. $N(M-1) \geq P-1$.

If $L = 1$, b must equal zero. Equation (14) will never be satisfied since a can only take on values in $\{0, 1, \ldots, P-1\}$, and c will not be an integer.

When $L \leq 3$ and $M = 2$, the system depends on h (the modulation index). The following examples demonstrate the above statement.

Example 1: $(M = 2, L = 2)$
Assume we have a sequence that satisfies $\sum_{i=N-L+1}^{N} \varepsilon_i = 0$. Equation (14) becomes
$$\frac{K}{P}(\frac{2a+b}{2}) = c.$$
Since $L = 2$ and $M = 2$, b can only take on values of 1 and -1, and c can be an integer only when P is odd and K is even.

Example 2: $(M = 2, L = 3)$

Equation (14) can be written as:

$$\frac{K}{P}\left(\frac{3a+b}{3}\right)=c.$$

Since $L=3$ and $M=2$, b can only take on values ± 1 or ± 2. In order for c to be an integer, K has to be a multiple of 3, and $3a+b$ has to be a multiple of P.

If $L \geq 2$ and $M > 2$ or $L > 3$ and $M = 2$, we can always find two input sequences that satisfy these conditions, and the system is catastrophic. Take a periodic sequence with period L, e.g., the sequence $\{\varepsilon_i\}_{i=0}^{\infty} = 0,\ldots,0,1,-2,1;0,\ldots,0,1,-2,1;\ldots$ where the semicolon denotes the begin of a new block of L symbols. If we pick N to be L, then this sequence satisfies both conditions (equations (12) and (13)), and the system is catastrophic.

3 Equivalent Non-Catastrophic Schemes

In this section we derive a canonical precoder such that any L-REC scheme in cascade with its precoder gives a non-catastrophic equivalent scheme. Let

$$\mathcal{U}=\{U_0,U_1,\cdots : U_i \in \{0,1,\cdots M-1\}\}$$

be the set of CPM input sequences. Let

$$\mathcal{W}_c=\{(\boldsymbol{U},\boldsymbol{U}') \in \mathcal{U}\times\mathcal{U} : D^2(\boldsymbol{U},\boldsymbol{U}')<\infty\}.$$

We are looking for a precoder which is a 1-1 correspondence $f:\mathcal{U}\rightarrow\mathcal{U}$ with the property that if $\boldsymbol{V},\boldsymbol{V}' \in \mathcal{U}$ and $(f(\boldsymbol{V}),f(\boldsymbol{V}')) \in \mathcal{W}_c$ then $\boldsymbol{V}$ and $\boldsymbol{V}'$ have finite Hamming distance. It is natural to require that f be linear over the ring of integers modulo M. Let $\boldsymbol{U}=f(\boldsymbol{V})$ and $\boldsymbol{U}'=f(\boldsymbol{V}')$. From the previous section we know that if $(\boldsymbol{U},\boldsymbol{U}') \in \mathcal{W}_c$ then for some positive integer N

$$\sum_{i=n-L+1}^{n}(U_i-U_i')=0, \text{ for all } n \geq N.$$

Clearly it is also true that

$$\begin{aligned} 0 &= R_M[\sum_{i=n-L+1}^{n}(U_i-U_i')] && (15)\\ &= R_M[\sum_{i=n-L+1}^{n}R_M[U_i-U_i']], \text{ for all } n \geq N. && (16)\end{aligned}$$

Now let $\Delta U_i = R_M[U_i - U_i']$ and $\boldsymbol{\Delta U} = \Delta U_0, \Delta U_1, \cdots$. Let $\boldsymbol{\Delta U}(D)$ be the D-transform of $\boldsymbol{\Delta U}$. The D-transform equivalent of (16) is $\boldsymbol{\Delta U}(D)(1 + D + D^2 + \cdots + D^{L-1}) = B(D)$, where $B(D)$ is a polynomial over the ring of integers modulo M of degree less that N. Hence

$$\boldsymbol{\Delta U}(D) = \frac{B(D)}{1 + D + D^2 + \cdots + D^{L-1}}.$$

On the other hand if $\boldsymbol{V}$ and $\boldsymbol{V}'$ have finite Hamming distance then $\Delta\boldsymbol{V} = \boldsymbol{V} - \boldsymbol{V}'$, where the difference is componentwise and modulo M, has a finite number of nonzero components. Equivalently, $\Delta\boldsymbol{V}(D)$ is a polynomial (i.e., it has finite degree). Due to the linearity of f it is enough to guarantee that if for some polynomial $B(D)$

$$\boldsymbol{U}(D) = \frac{B(D)}{1 + D + D^2 + \cdots + D^{L-1}}$$

is a precoder output sequence, the corresponding input sequence is also a polynomial. This is certainly true if the transfer function of the encoder is the D-transform

$$f(D) = \frac{1}{1 + D + D^2 + \cdots + D^{L-1}},$$

since then the input to $\boldsymbol{U}(D)$ is the polynomial $B(D)$. Figure 2 shows the general form of the precoder consisting of $L - 1$ shift registers. The cascade of the precoder with the CPE is shown in Figure 3. It is important to note that the number of states of the resulting equivalent CPE shown in Figure 3 is the same as the number of states of the original CPE in Figure 1. The two encoders are equivalent in the sense that both of them generate the same set of output sequences. The fact that they have the same number of states implies that they also have the same decoding complexity. However, when we combine the encoder in Figure 3 with the memoryless modulator we are guaranteed that the resulting CPM scheme is non-catastrophic. Of particular interest is the case when $M = P$ since then the encoder of Figure 3 is a convolutional encoder over the ring of integers modulo M.

References

[1] B. Rimoldi, 'A Decomposition Approach to CPM,' *IEEE Trans. Inform. Theory,* vol. 34, No. 2, Mar. 1988.

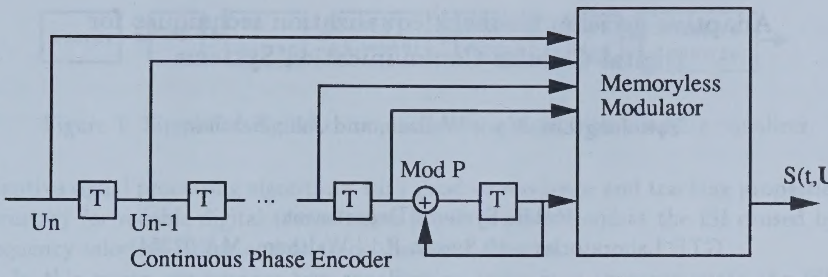

Memoryless
Modulator
Mod P
S(t,U)
T
T
T
T
...
Un
Un-1
Continuous Phase Encoder

Figure 1. A general description of CPM.

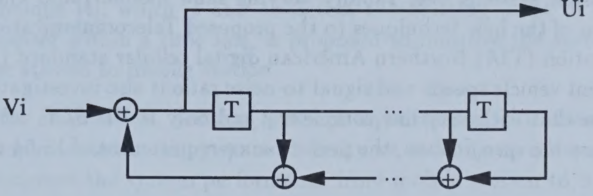


Figure 2. Precoder - The number of shift registers is L-1. Addition is modulo M.

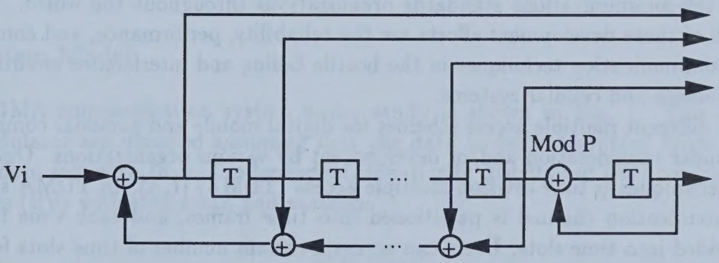

Mod P
Vi
T
T
...
T
T
...

Figure 3. Equivalent CPE.

Adaptive decision feedback equalization techniques for Digital Cellular Communication Systems

Yow-Jong Liu, Mark Wallace, and John Ketchum

Mobile Systems Department
GTE Laboratories - 40 Sylvan Rd., Waltham, MA 02254

Abstract

In this paper, two adaptive equalization techniques, diversity reception and bi-directional equalization, are proposed to improve the performance of digital transmission systems over rapidly varying land mobile radio channels. Application of the new techniques to the proposed Telecommunications Industry Association (TIA) Northern American digital cellular standard (IS-54) under different vehicle speeds and signal-to-noise ratio is also investigated. The results show that with very low complexity and only 16 dB S/N, *which is 6 dB lower than the specification*, the performance requirement of IS-54 can be met.

1 Introduction

Highly spectrum-efficient digital transmission methods for terrestrial mobile and personal communication systems are currently being investigated in industrial research laboratories and telecommunications standards organizations throughout the world. Critical issues in all of these development efforts are the reliability, performance, and complexity of digital communication techniques in the hostile fading and interference environments typical of mobile and cellular systems.

Several different multiple access schemes for digital mobile and personal communications are under consideration and/or development by various organizations. One of the proposed techniques is time division multiple access (TDMA) [1, 2]. In TDMA systems, every communication channel is partitioned into time frames, and each time frame is further divided into time slots. Users can access a certain number of time slots for voice or data communication.

The mobile radio communication channel in urban and suburban areas is typically modeled as a frequency selective multipath Rayleigh fading channel subject to log-normal shadowing. During arbitrary TDMA frame and slot durations (typically 20—40 msec, and 6—7 msec), the channel characteristics may vary significantly over a time slot for rapidly moving vehicles, or may appear to be stationary (slow variation) over many time slots for slowly moving vehicles or pedestrians using a hand-held unit. Further, the delay spread on the channel may vary between a fraction of a symbol in duration to one or two symbols in duration, in extreme cases. This time dispersive nature of the channel results in severe intersymbol interference (ISI), which significantly degrades the digital transmission system performance if left uncompensated. Equalization and

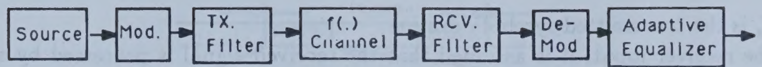


Figure 1: Simplified digital communication system with adaptive equalizer.

adaptive signal processing algorithms with good convergence and tracking properties are necessary for reliable digital transmission [3], in order to combat the ISI caused by the frequency-selective fading and to track the rapid channel.

In this paper, we propose two equalization techniques to compensate the ISI and channel variation in the proposed Telecommunications Industry Association (TIA) next generation North American digital cellular standard (IS-54):

1. In the forward-link, a new bidirectional adaptive decision feedback equalization technique [4], which is able to detect the starting point of the divergence of the equalizer within a time slot, is proposed to improve the system performance from base station to mobile station.
2. In the reverse-link, decision feedback equalization with diversity reception, which is implemented by using space diversity, or antenna diversity reception, is proposed to improve the system performance from mobile station to base station.

The rest of this paper is organized as follows. Section 2 describes the system model and the IS-54 TDMA frame structure used in this study. Section 3 describes in detail the proposed techniques for both forward-link and reverse-link communication in IS-54. Section 4 contains numerical results and discussion of complexity issues. Finally the conclusions of this study are given in section 5.

2 System Model

The TDMA communication system under study is shown in Fig. 1. The source and the modulator are modeled assuming that the data to be transmitted forms a stationary random sequence $\{a_k\}$ of independent, identically distributed (iid) complex random variables (RV) with zero mean and variance

$$\sigma_b^2 \triangleq \mathcal{E}|a_k|^2. \tag{1}$$

The data sequence $\{a_k\}$ is sent to a linear modulator. It is modeled as the cascade of a modulator having the ideal impulse $\delta(t)$ as its basic waveform and a shaping filter with an impulse response $p(t)$ and frequency response $\mathcal{P}(f)$. Thus, the modulated signal is

$$\sum_{k=-\infty}^{\infty} a_k\delta(t-kT_s) \tag{2}$$

and the signal sent to the channel is

$$s(t) = \sum_{k=-\infty}^{\infty} a_k p(t-kT_s), \tag{3}$$

where T_s is the transmitted symbol duration.

At the receiver input, it is assumed that the received signal is processed by a filter matched to the transmitted symbol pulse shape, $p(t)$. Assuming perfect receiver synchronization, and as a result of assuming T_s-spaced path delays, we can write the sampled output of the symbol pulse matched filter at time nT_s as

$$x_k = \sum_{m=0}^{L} f_m(k) a_{k-m} + n_k, \; k \geq 0 \tag{4}$$

where $f_m(k)$, $1 \leq m \leq L$, $k \geq 0$ are L zero mean discrete time complex Gaussian random processes, uncorrelated in m, but generally correlated in k. The autocorrelation function for each random process $f_m(\cdot)$, assuming a uniform scattering model, is given by [3]

$$R(\tau) = A J_0(2\pi f_d \tau), \tag{5}$$

where A is some normalized constant and f_d is maximum Doppler frequency, determined by the maximum vehicle speed and carrier frequency. For example, the maximum Doppler frequency, f_d, associated with a vehicle traveling at 60 mph is about 80 Hz at 900 MHz carrier frequency.

The noisy, distorted symbol sequence $\{x_k\}$ is then passed to the adaptive equalizer. The purpose of the adaptive equalizer is to track the channel variation and compensate for the time-varying time dispersive nature of the channel. A primary goal in the design of the equalizer is to realize the potential for a diversity gain by resolving the multipath structure of the channel, so as to recover the transmitted symbol sequence $\{a_k\}$.

2.1 TDMA channel structure

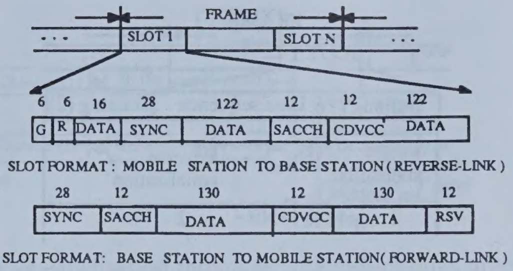
In TDMA systems, every communication channel is partitioned into time frames, and each time frame is further divided into time slots of some finite period of N_s symbols duration. Users can access a certain number of time slots for voice or data communication. The TDMA frame structure adopted by the TIA is shown in Fig. 2.

3 Adaptive Equalization Techniques

Performance of digital communication systems in the presence of Rayleigh fading is limited by the occurrence of deep fades which cause the instantaneous signal-to-noise ratio to approach zero, and the instantaneous bit error probability to approach one half. When the Rayleigh fading is compounded by time dispersion requiring adaptive equalization, the occurrence of a deep fade causes the equalizer to diverge, since either there is no signal available to guide the adaptation algorithm, or the channel variation is too fast for the adaptive equalizer to track. Typically after such an event, the equalizer cannot recover until a sequence of training symbols is transmitted. In the TDMA systems considered here, this means that the equalizer cannot recover until the beginning of the next time slot, which begins with a known training sequence.

3.1 Proposed Equalization Techniques for forward-link communication

From section 2, we know that the base station will continuously transmit one of the six possible training sequences for each time slot no matter whether there is an active mobile


FRAME
SLOT 1
SLOT N
6 6 16 28 122 12 12 122
G R DATA SYNC DATA SACCH CDVCC DATA
SLOT FORMAT : MOBILE STATION TO BASE STATION (REVERSE-LINK)
28 12 130 12 130 12
SYNC SACCH DATA CDVCC DATA RSV
SLOT FORMAT: BASE STATION TO MOBILE STATION(FORWARD-LINK)

NOTE: All number indicate bits. (Two bits p er symbol)
G - Guard Time
R - Ramp Time
SACCH - Slow Associated Control Channel
SYNC - Synchronization and Trai ning
DATA - Traff i c Chann el
CDVCC - Coded Digital Voice Color Code
RSV - Reserved for Future Use

Figure 2: IS-54 TDMA frame structure.

user assigned to each particular time slot or not. Based on this information, we propose a new equalization technique: *bi-directional equalization*. The bi-directional equalization technique introduces a new equalization operation for TDMA digital communication systems, which is shown in Fig. 3. All the received symbols in the current *effective time slot* are stored in a temporary buffer. [1] The major difference between our approach and conventional techniques is that an additional control mechanism is added to monitor the error signal which provides information on the tracking status of the equalizer. The algorithm, which is shown in Fig. 4, of the new approach is listed below:

1. Start forward equalization and monitor the error signals between the estimated symbols and quantized symbols. If $\sum |error\ signal| \geq thresholdA$, stop at this

---

[1] An effective time slot contains all the training and data symbols in the current time slot and the training symbols only in the adjacent time slot.

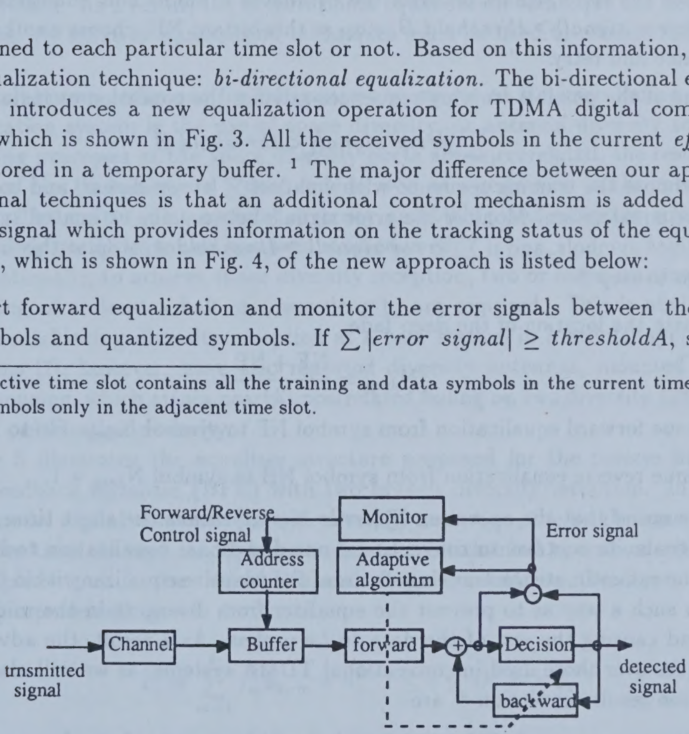

Forward/Reverse Control signal
Monitor
Error signal
Address counter
Adaptive algorithm
Channel
Buffer
forward
Decision
backward
trnsmitted signal
detected signal

Figure 3: Bi-directional equalization technique with the capability of detecting starting point of the divergence of the tracking algorithm within a given time slot

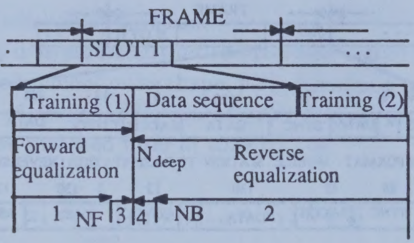


Figure 4: Bi-directional equalization procedure for TDMA communication systems with available adjacent training symbols

instant NF, and go to step 2.

2. Choose any one of the possible training sequences and start reverse equalization. Monitor the error signals between the estimated symbols and quantized symbols. If $\sum |error\ signal| \geq threshold\ B$, stop at this instant NR, choose another training sequence and retry.

   If none of the possible training sequences satisfies the condition at instant NR,

   $$\sum |error\ signal| < threshold\ B$$

   then choose the training sequence with smallest $\sum |error\ signal|$ and continue the reverse equalization. Monitor the error signals between the estimated symbols and quantized symbols, and if $\sum |error\ signal| \geq threshold\ A$, stop at this instant NB, and go to step 3.

3. Estimate the location of the deep fade:

   $$\mathrm{N_{deep}} = \frac{\mathrm{NF} + \mathrm{NB}}{2}. \tag{6}$$

   Continue forward equalization from symbol NF to symbol $\mathrm{N_{deep}}$. Go to step 4.

4. Continue reverse equalization from symbol NB to symbol $\mathrm{N_{deep}} + 1$.

It should be noted that the operation $\sum |error\ signal|$ means the short time average of the error signals. In contrast to conventional one-directional equalization techniques [5] our technique can estimate the tracking status of the adaptive equalizer within the current time slot in such a way as to prevent the equalizer from diverging in the middle of the time slot and causing the rest of the data to be useless. As a result, the advantages of our techniques over those used in conventional TDMA systems, as we will also see from the simulation results in section 4, are:

1. The divergence period of the equalizer is limited to a very short time and near the occurrence of the deep fade only.
2. The long error bursts which occur after the divergence of the equalizer are minimized. Thus, the complexity of interleaver/deinterleaver is reduced and so is the delay caused by the interleaving/deinterleaving operation.
3. The system performance is better and can be achieved with a lower signal power requirement.

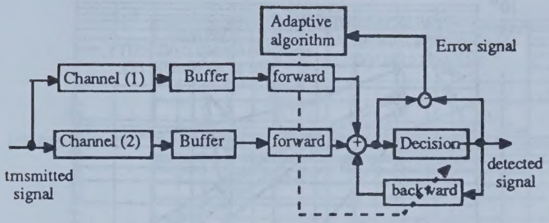


Figure 5: Adaptive equalization technique with two-channel diversity reception.

3.2 Proposed Equalization Techniques for reverse-link communication

From section 2, we know that the training symbols are available to the base station for any particular time slot only when there is an active mobile user using that particular time slot. In other words, not all the adjacent time slots can provide training symbols to the base station. As a result, the bi-directional equalization technique can not be applied to this case. Alternative equalization techniques must be used to combat fading and ISI problems.

One well-known technique for mitigating the effects of Rayleigh fading in a mobile communication system is the use of space diversity, or antenna diversity reception [3]. If the fading processes at the space diversity ports are uncorrelated, the result will be a reduction of the probability of a deep fade as well as a reduction in the rate of channel variation. The use of joint optimal diversity combining and adaptive equalization to combat fading on HF channels has been studied in [5].

Conventionally, to achieve space diversity reception, two or more antennas, mounted at a spacing of at least a half of a wavelength, are required. This is obviously not a problem for achieving diversity reception at a base station in a cellular system. Recent publications [7], however, have also reported diversity antennas, mounted in a single compact housing, which attain nearly uncorrelated fading on two diversity paths, through the use of polarization diversity.

Figure 5 illustrates the equalizer structure proposed for the reverse link. It is a decision feedback equalizer (DFE) with two branch diversity reception. In general, in order to model digital transmission with diversity reception, the single input/single output discrete-time channel model is extended to multichannel operation with single input/multiple output. In the case of D-branch diversity reception, the D received signals, x_k^ℓ, can be represented by

$$x_k^\ell = \sum_{m=1}^{L} f_m^\ell b_{k-m} + n_k^\ell, \; 1 \le \ell \le D. \; k \ge 0, \tag{7}$$

where $\{f_m^\ell\}$ are the tap coefficients at time kT_s for the m-th delay on the ℓ-th channel, and n_k^ℓ is the additive white Gaussian noise sequence on the ℓ-th channel. The received signals x_k^ℓ, $1 \le \ell \le D$, are the inputs to the D feedforward filters in the diversity receiver, and are maximal-ratio combined by the M-input feedforward section of the diversity DFE. A single feedback filter is used to suppress residual intersymbol interference associated with previously detected symbols, as shown in Figure 5. The estimated symbol at time

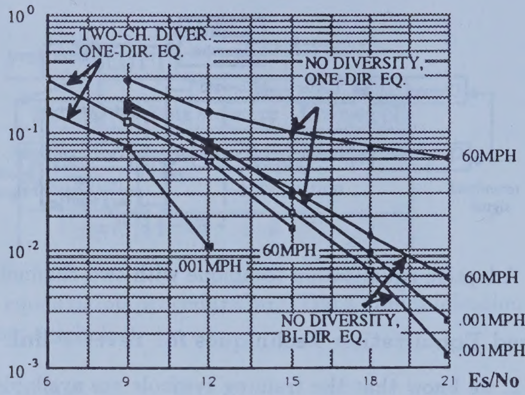


Figure 6: BER vs. Es/No for different vehicle speeds with one-directional/bi-directional equalization with and without diversity reception.

k is

$$y_k = \sum_{\ell=1}^{D} \sum_{i=-M_1}^{0} \alpha_i^{\ell} x_{k-i} - \sum_{j=1}^{M_2} \beta_j \hat{y}_{k-j}. \tag{8}$$

where $\hat{y}_k$ is the estimated symbol after decision device, α_i are the feedforward tap coefficients, and β_j are the feedback tap coefficients. In (8), D is 2 for two channel diversity reception.

4 Simulation Results and Discussion

In this section, we present the applications of the proposed techniques described previously to the next generation North American Digital Cellular Communication System. The modulation scheme is $\pi/4$-shift QDPSK. The TDMA frame structure is the same as proposed in IS-54. The frequency-selective mobile radio channel is modeled as a two-ray model with two equal strength, independently Rayleigh fading paths and one symbol interval between two paths. Recursive least-squares (RLS) type algorithms are chosen to update the equalizer tap coefficients.

4.1 Forward-link communication

Bi-directional equalization without diversity reception is used in this communication link. The number of feedforward and feedback taps and forgetting factor are chosen to be 3,1, and .85, respectively.

Fig. 6 illustrates the simulated BER performance of the bi-directional equalization technique for the two-ray model at different vehicle speeds. It also includes the results of the one-directional equalization technique for comparison. The new technique provides more than 10dB energy gain over the conventional one-directional equalization technique

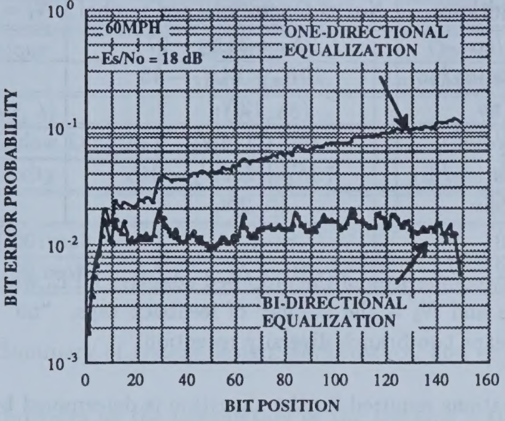


Figure 7: BER as a function of bit position in a given time slot with one-directional and bi-directional equalizer.

at 5% BER, and requires an Es/No of only 16 dB, *which is 6 dB lower than the specification in the IS-54 standard*, to achieve 3% BER at a vehicle speed of 60 mph.

Fig. 7 shows the BER as a function of bit position in a given time slot. We can observe that in the case of the one-directional equalization technique, BER is an increasing function of bit position. This can be explained as follows. Divergence of the equalizer at the beginning of the time slot will affect the correctness of the received symbols at the end of the time slot. However, divergence of the equalizer at the end of the time slot will not affect the correctness of the received symbols at the beginning of the time slot.

From the previous discussion, we know that the bi-directional equalization technique can estimate the tracking status of the equalizer so as to limit the divergence of the equalizer to a very short period near the occurrence of the deep fade only. As a result, very widely uniform BER except the region near the beginning of the tracking mode can be achieved by the bi-directional equalization technique in a given time slot.

4.2 Reverse-link communication

Decision feedback equalization with two channel diversity reception is used in this link. The forgetting factor λ is chosen to be 0.95.

Fig. 6 also illustrates the BER performance of the one-directional equalization technique with and without diversity reception for the two-ray model at different vehicle speeds. The simulation results show that two-channel diversity provides more than 10 dB energy gain over the case without diversity reception, at a bit error rate of 5%.

4.3 Complexity Issues

The complexity of the proposed techniques in terms of the number of additions, multiplications, and divisions per iteration is shown in Table 1. It should be noted that

RLS algorithm	No. of Operations per symbol	$N_1 = 3, N_2 = 1$
Addition/no	$9N_1 + 34N_2 - 27$	20
Multiplication/no	$19N_1 + 44N_2 - 45$	56
Division/no	$2N_1$	6
Addition/div	$43N_1 + 116N_2 - 130$	115
Multiplication/div	$73N_1 + 206N_2 - 245$	180
Division/div	$2N_1$	6

Table 1: Complexity comparison of the proposed adaptive DFE, where N_1 is the number of feedforward taps and N_2 is the number of feedback taps. "no" means no diversity reception. "div" means two-branch diversity reception.

the number of operations required for the algorithm is determined by assuming that the multichannel lattice structure is used [6] .

The total number of million of instructions per second (MIPS) executed by the DSP processor can be calculated as follows:

$$(\text{addition} + \text{multiplication} + \text{division}) \times \text{effective symbols/slot} \times \text{slot/second} \times 10^{-6} \quad (9)$$

The effective symbols per slot should include the training symbols in the adjacent time slot for every possible case in the case of forward-link communication.

In general, addition and multiplication operations can be achieved within one instruction cycle by most state-of-the-art digital signal processors. If we further assume that one division operation can be finished within ten instruction cycles then the number of MIPS required per second can be calculated as follows:

- **forward-link:**

$$(20 + 56 + 10 \times 6) \times 246 \text{ symbols/slot} \times 50 \text{ slot/second} \times 10^{-6} = 1.6728\text{MIPS}. \quad (10)$$

- **reverse-link:**

$$(115 + 180 + 10 \times 6) \times 162 \text{ symbols/slot} \times 50 \text{ slot/second} \times 10^{-6} = 2.8755\text{MIPS}. \quad (11)$$

From (10) and (11), we know that about 2 MIPS is executed by the equalization operation in the mobile stations and about 3 MIPS is executed in the base stations, which can be easily achieved by most available DSP devices.

5 Conclusions

In this paper, we present an investigation of the equalization issue in the proposed next generation North American digital cellular standard (IS-54).

We propose two low-complexity adaptive equalization techniques to improve the overall digital transmission system performance over time varying land mobile radio channels.

In the forward-link, we suggest the using of a newly developed bidirectional decision feedback equalization technique. The major difference between our approach and conventional techniques is that an additional control mechanism is added to monitor the

	FORWARD-LINK	REVERSE-LINK
Technique	Bi-directional Equalization	One-dir. EQ. with two-channel div. reception
(N1, N2, λ)	(3,1,.85)	(3,1,.95)
(Threshold, Window Length)	(.5, 9)	Not Available
Complexity	2MIPS	3MIPS
	Es/No	Es/No
3 % BER (.001mph)	14 dB	10.5 dB
3 % BER (60 mph)	15.5 dB	12.5 dB

Table 2: Summary of system design, performance, and complexity.

error signal, which provides us the information of the tracking status of the equalizer, and is used to detect the starting point of the divergence of the equalizer within a given time slot.

In the reverse-link, we suggest the use of conventional one-directional decision feedback equalization with two channel diversity reception, which is implemented by using antenna diversity or polarization diversity reception techniques.

Table 2 shows the summary of the complete equalizer design, performance, and complexity.

Acknowledgements
The authors want to thank Dr. D. Fye for his encouragement of this work.

References

[1] M. R. L. Hodges, "The GSM Radio Interface," *British Telecom Technology,* Vol. 8, No. 1, pp.31-43, January 1990.

[2] TR45.3 IS-54, "Dual-Mode Mobile Station - Base Station Compatibility Standard."

[3] W. Lee, *Mobile Communications Engineering.* McGraw-Hill Book Company, 1982.

[4] Y. J. Liu, "Bi-directional Equalization Technique for TDMA Communication Systems," submitted for U. S. pattern application.

[5] P. Monsen, "Theoretical and measured performance of a DFE modem on a fading multipath channel," *IEEE Trans., COM-25,* pp. 1144-1153, 1977.

[6] F. Ling and J. G. Proakis, "Adaptive Lattice Decision-Feedback Equalizers - Their Performance and Application to Time-variant Multipath Channels," *IEEE Trans. Comm.,* Vol. COM-33, No. 4, pp.348-356, April 1985.

[7] H. Kuboyama, Y. Tanaka, K. Sato, K. Fujimoto, and K. Hirasawa, "Experimental Results with Mobile antennas Having Cross-Polarization Components in Urban and Rural Areas," IEEE Trans. Vehicle Tech. VOL. 39, No. 2, pp150-160, May 1990.

CHARACTERIZATION OF ALL CLOSED-LOOP TRANSFER FUNCTION MATRICES IN H_∞-OPTIMIZATION

Anton A. Stoorvogel [1]
Dept of Electrical Engineering
and Computer Science
University of Michigan
Ann Arbor, MI 48109-2122

Ali Saberi [2]
School of Electrical Engineering
and Computer Science
Washington State University
Pullman, WA 99164-2752

Ben M. Chen [2]
School of Electrical Engineering
and Computer Science
Washington State University
Pullman, WA 99164-2752

ABSTRACT

In this paper we derive a characterization of all stable closed loop systems with H_∞ norm strictly less than 1 which we can obtain via a suitable stabilizing feedback. We give an exact characterization. However this characterization contains relatively implicit constraints on the free parameter. We also introduce an "approximate" characterization parameterized via a stable system X with H_∞-norm less than 1 (and no other conditions on X). A element of this approximate characterization can be arbitrarily well approximated by a closed loop system we can obtain via a suitable stabilizing feedback.

1. INTRODUCTION

In H_∞ control (see e.g. [1,4,6]) it is well-known that suitable controllers are not unique. This is in part because we in general investigate suboptimal design (make the H_∞ norm less than some a priori given number γ) and also because even optimal controllers are in the MIMO case non-unique.

An interesting question one might therefore ask is the following: characterize all closed loop systems with H_∞ norm less than γ that we can obtain via a suitable stabilizing feedback. In several papers (see e.g. [1,6]) a characterization is given of all time-invariant controllers which stabilize a given linear time-invariant system and result in a closed loop system with H_∞ norm strictly less than γ. This can be used in a straightforward manner to characterize the closed loop systems these controllers generate. However, this is done under some assumptions on the direct feedthrough matrices of the system (the so-called regular case). Without these conditions (the singular case) necessary and sufficient conditions for the existence of a suitable controller are available (see [4]). On the other hand, for this singular case relatively little is known about closed loop systems one can obtain.

---

[1] On leave from the Dept. of Mathematics, Eindhoven University of Technology, the Netherlands. The research of Dr. A.A. Stoorvogel has been made possible by a fellowship of the Royal Netherlands Academy of Arts and Sciences and he is also supported by the Netherlands Organization for Scientific Research (NWO).

[2] The work of A. Saberi and B. M. Chen is supported in part by Boeing Commercial Airplane Group and in part by NASA Langley Research Center under grant contract NAG-1-1210.

In this paper we derive a characterization of all stable closed loop systems with H_∞ norm strictly less than 1 which we can obtain via applying a suitable stabilizing feedback to the given system Σ. We can replace 1 by γ via simple scaling. The closed-loop systems are parameterized via a stable system X with H_∞-norm less than 1. However, these systems X have to satisfy two other, relatively implicit, extra conditions. Therefore we also give an approximate characterization. It is the same characterization except that X does not have to satisfy these extra two conditions; the system X is an arbitrary stable system with H_∞ norm strictly less than 1. The trade-off is that it is an approximate characterization. For each stable system X with H_∞ norm less than 1 we generate a system which can be arbitrarily well approximated with a closed loop system which we obtain by applying a suitable stabilizing controller to our system Σ. Conversely any closed loop system with H_∞ norm strictly less than 1, which we can obtain by applying a stabilizing controller to Σ, is identical to a system we obtain for a suitable choice of the parameter X in our characterization.

In other words, we find a simpler characterization of the "closure" (the approximate set is not actually closed but lies between the set itself and its closure) of the set of attainable closed loop systems. Finally we would like to note that this approximate characterization and the actual characterization are equal in the regular case.

The formal problem statement will be given in the next section. In section 3 we will recall some preliminary results. In section 4 we will give an exact characterization of all closed loop systems. Finally, in section 5 we give the, much simpler approximate characterization. We conclude with some final remarks in section 6.

2. PROBLEM STATEMENT

We consider the linear, time-invariant, finite-dimensional system:

$$\Sigma : \begin{cases} \dot{x} = Ax + Bu + Ew, \\ y = C_1 x + D_1 w, \\ z = C_2 x + D_2 u, \end{cases} \tag{2.1}$$

where $x \in \Re^n$ is the state, $u \in \Re^m$ is the control input, $w \in \Re^\ell$ is the unknown disturbance, $y(t) \in \Re^p$ is the measured output and $z \in \Re^q$ is the unknown output to be controlled. A, B, E, C_1, C_2, D_1 and D_2 are matrices of appropriate dimensions. The following assumptions are made:

(a) (A, B) is stabilizable and (A, B, C_2, D_2) has no invariant zeros on the $j\omega$ axis, and

(b) (A, C_1) is detectable and (A, E, C_1, D_1) has no invariant zeros on the $j\omega$ axis.

Throughout this paper we will assume that there exists an internally stabilizing controller of the form:

$$\Sigma_F : \begin{cases} \dot{v} = Kv + Ly, \\ u = Mv + Ny, \end{cases} \tag{2.2}$$

such that the H_∞-norm of the closed-loop transfer function from z to w, $T_{zw}(s)$, is strictly less than 1. To be precise, let us define the sets

$$\mathbf{C} := \{ \Sigma_F \mid \Sigma \times \Sigma_F \text{ is stable and } \|T_{zw}\|_\infty < 1 \} \tag{2.3}$$

and

$$\mathbf{T} := \{ T_{zw} \mid \Sigma_F \in \mathbf{C} \}. \tag{2.4}$$

Elements of the sets $\mathbf{C}$ and $\mathbf{T}$ will sometimes be called suitable controllers and suitable closed loop systems respectively. The goal of this paper is to characterize the set $\mathbf{T}$, i.e. all the closed-loop transfer function $T_{zw}(s)$ satisfying $\|T_{zw}(s)\|_\infty < 1$.

3. PRELIMINARY

In this section, we recall some results from [4]. A central role in our study of the above problem will be played by the quadratic matrix inequality. For matrix $P \in \Re^{n \times n}$ we consider the following matrix:

$$F(P) := \begin{bmatrix} A^{\mathrm{T}}P + PA + C_2^{\mathrm{T}}C_2 + PEE^{\mathrm{T}}P & PB + C_2^{\mathrm{T}}D_2 \\ B^{\mathrm{T}}P + D_2^{\mathrm{T}}C_2 & D_2^{\mathrm{T}}D_2 \end{bmatrix}.$$

If $F(P) \geq 0$, we say that P is a solution of the quadratic matrix inequality.

We also define a dual version of this quadratic matrix inequality. For $Q \in \Re^{n \times n}$ we define the following matrix:

$$G(Q) := \begin{bmatrix} AQ + QA^{\mathrm{T}} + EE^{\mathrm{T}} + QC_2^{\mathrm{T}}C_2Q & QC_1^{\mathrm{T}} + ED_1^{\mathrm{T}} \\ C_1Q + D_1E^{\mathrm{T}} & D_1D_1^{\mathrm{T}} \end{bmatrix}.$$

If $G(Q) \geq 0$, we say that Q is a solution of the dual quadratic matrix inequality. In addition to these two matrices, we define two matrices pencils, which play dual roles:

$$L(P,s) := \begin{bmatrix} sI - A - EE^{\mathrm{T}}P & -B \end{bmatrix},$$

$$M(Q,s) := \begin{bmatrix} sI - A - QC_2^{\mathrm{T}}C_2 \\ -C_1 \end{bmatrix}.$$

Finally, we define the following two transfer matrices:

$$G_{ci}(s) := C_2 (sI - A)^{-1} B + D_2,$$

$$G_{di}(s) := C_1 (sI - A)^{-1} E + D_1,$$

Let $\rho(M)$ denote the spectral radius of the matrix M. Then the following theorem characterizes the existence of suitable controllers.

Theorem 3.1. *Consider the system (2.1). Assume that both the system (A, B, C_2, D_2) as well as the system (A, E, C_1, D_1) have no invariant zeros on the imaginary axis. Then the following two statements are equivalent:*

1. *For the system (2.1) there exists a time-invariant, finite-dimensional dynamic compensator Σ_F of the form (2.2) such that the resulting closed-loop system, with transfer matrix $T_{zw}(s)$, is internally stable and has H_∞ norm less than 1, i.e. $\|T_{zw}(s)\|_\infty < 1$.*
2. *There exist positive semi-definite solutions P, Q of the quadratic matrix inequalities $F_\gamma(P) \geq 0$ and $G(Q) \geq 0$ satisfying $\rho(PQ) < 1$ such that the following rank conditions are satisfied*
    - *(a) rank $F(P) = \text{rank}_{\mathcal{R}(s)} G_{ci}$,*
    - *(b) rank $G(Q) = \text{rank}_{\mathcal{R}(s)} G_{di}$,*
    - *(c) rank $\begin{bmatrix} L(P,s) \\ F(P) \end{bmatrix} = n + \text{rank}_{\mathcal{R}(s)} G_{ci}, \ \forall\, s \in \mathcal{C}^0 \cup \mathcal{C}^+$,*
    - *(d) rank $\begin{bmatrix} M(Q,s) & G(Q) \end{bmatrix} = n + \text{rank}_{\mathcal{R}(s)} G_{di}, \ \forall\, s \in \mathcal{C}^0 \cup \mathcal{C}^+$.* □

Our goal is to characterize the set of all closed loop systems with H_∞ norm less than 1 which we can obtain by applying a suitable stabilizing controller. By the above theorem, this set is empty if the conditions in part 2 are not met. Therefore in the remainder of this paper we will assume that there exist matrices P and Q satisfying the conditions in part 2 of the above theorem. We can now start with the derivation of the characterization of all suitable closed loop systems.

Next, we construct a new system,

$$\Sigma_{P,Q} : \begin{cases} \dot{x}_{P,Q} = A_{P,Q} x_{P,Q} + B_{P,Q} u_{P,Q} + E_{P,Q} w, \\ y_{P,Q} = C_{1,P} x_{P,Q} \qquad\qquad\qquad + D_{P,Q} w, \\ z_{P,Q} = C_{2,P} x_{P,Q} + D_P u_{P,Q}, \end{cases} \tag{3.1}$$

where

$$F(P) = \begin{bmatrix} C_{2,P}^{\mathrm{T}} \\ D_P^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} C_{2,P} & D_P \end{bmatrix}, \qquad G(Q) = \begin{bmatrix} E_Q \\ D_{P,Q} \end{bmatrix} \begin{bmatrix} E_Q^{\mathrm{T}} & D_{P,Q} \end{bmatrix},$$

such that $\begin{bmatrix} C_{2,P} & D_P \end{bmatrix}$ and $\begin{bmatrix} E_Q^{\mathrm{T}} & D_{P,Q} \end{bmatrix}$ are both surjective. Moreover,

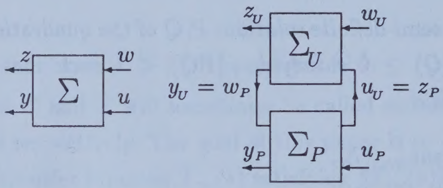
$$\begin{aligned} A_{P,Q} &:= A + EE^{\mathrm{T}}P + (I - QP)^{-1} Q C_{2,P}^{\mathrm{T}} C_{2,P}, \\ B_{P,Q} &:= B + (I - QP)^{-1} Q C_{2,P}^{\mathrm{T}} D_P, \\ E_{P,Q} &:= (I - QP)^{-1} E_Q, \\ C_{1,P} &:= C_1 + D_1 E^{\mathrm{T}} P. \end{aligned}$$

It has been shown in [4] that this new system has the following properties:

1. $(A_{P,Q}, B_{P,Q}, C_{2,P}, D_P)$ is right invertible and minimum phase.

2. $(A_{P,Q}, E_{P,Q}, C_{1,P}, D_{P,Q})$ is left invertible and minimum phase.

In [4] the transformation to $\Sigma_{P,Q}$ is done in two stages. In the first stage (the transformation into a system Σ_P) a system Σ_U is constructed which connects Σ and Σ_P, i.e. the following systems have the same realization except for some extra stable uncontrollable dynamics on the right hand side



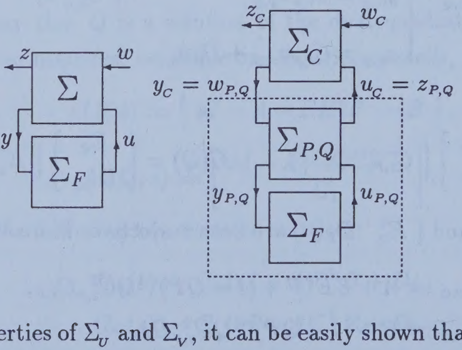
(3.2)

Moreover, this system Σ_U is inner, i.e the system is stable and the transfer function of Σ_U from (w_U, u_U) to (z_U, y_U), say G_U, has the following property:

$$G_U^{\mathrm{T}}(-z)G_U(z) = G_U(z)G_U^{\mathrm{T}}(-z) = I,$$

for any $z \in C$ which is not a pole of the system $G_U(s)$. Finally, the subsystem from w_U to y_U has a stable inverse. Similarly, we can connect Σ_P and $\Sigma_{P,Q}$ via some system Σ_V via a dual argument.

In this way we can derive that the original system Σ in (2.1) and the new system $\Sigma_{P,Q}$ have a similar connection. In other words, we can construct a system Σ_C (which is simply the interconnection of Σ_U and Σ_V) such that the following two interconnections have the same realization for every controller Σ_F except for some extra stable uncontrollable or unobservable dynamics on the right hand side:



(3.3)

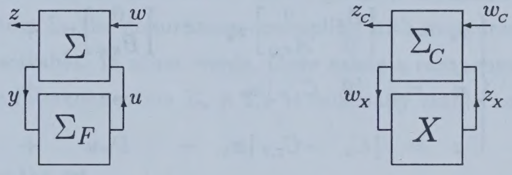
Due to the properties of Σ_U and Σ_V, it can be easily shown that Σ_C is inner. Moreover, by applying Redheffer's lemma (see [1]) and its dual version we can derive the following theorem:

Theorem 3.2. *For any given compensator Σ_F of the form (2.2) the following two statements are equivalent:*

(i) *Σ_F applied to the system Σ defined by (2.1) is internally stabilizing and the resulting closed loop transfer function from w to z has H_∞-norm less than 1, i.e. $\|T_{zw}(s)\|_\infty < 1$.*

(ii) *Σ_F applied to the new system $\Sigma_{P,Q}$ defined by (3.1) is internally stabilizing and the resulting closed loop transfer function from $w_{P,Q}$ to $z_{P,Q}$, $T_{z_{P,Q}w_{P,Q}}(s)$, has H_∞ norm less than 1, i.e. $\|T_{z_{P,Q}w_{P,Q}}(s)\|_\infty < 1$.* □

Next, we denote the system inside the dashed box of (3.3) by $X(s)$. We can then simplify the picture:



(3.4)

Our goal of this paper is to characterize all suitable closed-loop systems $\Sigma \times \Sigma_F$ as in the left of (3.4), i.e. the set $\mathbf{T}$ as defined in the previous section. By the previous theorem if the closed loop system on the left in (3.4) is stable and has H_∞ norm strictly less than 1 then X, defined to be equal to the dashed box in (3.3), is stable and has H_∞ norm strictly less than 1. Our goal is to show the "converse": for any stable system X with H_∞ norm strictly less than 1 the interconnection on the right hand side of (3.4) is asymptotically stable and has H_∞ norm strictly less than 1. Moreover, we can find a system Σ_F such that the two interconnections in (3.4) are both stable and arbitrarily close in H_∞ norm. In the next section we will show for which systems X we can make the interconnections equal. In section 5 we show that for all strictly proper X which are stable and have H_∞ norm strictly less than 1 we can always make the interconnections arbitrarily close in H_∞ norm.

We would like to conclude this section by stressing that the construction of Σ_C is an straightforward application of the results in [4]. It is only because of space limitations that we do not give this explicit construction in this paper.

4. EXACT CHARACTERIZATION

In this section, we will characterize the set $\mathbf{T}$ defined in (2.4). We first give the following result which is a straightforward application of the results in the previous section.

Lemma 4.1. *Let X be a stable system described by*

$$X : \begin{cases} \dot{x}_x = A_x x_x + B_x w, \\ z = C_x x_x + D_x w, \end{cases} \tag{4.1}$$

where A_x is stable and the transfer matrix of X has H_∞ norm strictly less than 1. Then the interconnection $\Sigma_C \times \Sigma_X$ as given on the right hand side in (3.4) is internally stable and the resulting closed-loop transfer functionfrom w_C to z_C has H_∞-norm less than 1. □

For a system X satisfying the conditions of the above lemma we define the following auxiliary system:

$$\Sigma_a : \begin{cases} \dot{x}_a &= \begin{bmatrix} A_x & 0 \\ 0 & A_{P,Q} \end{bmatrix} x_a &+ \begin{bmatrix} 0 \\ B_{P,Q} \end{bmatrix} u &+ \begin{bmatrix} B_x \\ E_{P,Q} \end{bmatrix} w, \\ y &= [0 \quad C_{1,P}]\, x_a & &+ D_{P,Q} w, \\ z &= [C_x \quad -C_{2,P}]\, x_a &- D_P u &+ D_x w, \end{cases} \tag{4.2}$$

For economy of notation, let us define

$$\tilde{A} = \begin{bmatrix} A_x & 0 \\ 0 & A_{P,Q} \end{bmatrix}, \quad \tilde{B} = \begin{bmatrix} 0 \\ B_{P,Q} \end{bmatrix}, \quad \tilde{E} = \begin{bmatrix} B_x \\ E_{P,Q} \end{bmatrix},$$

and

$$\tilde{C}_1 = [0 \quad C_{1,P}], \quad \tilde{D}_1 = D_{P,Q}, \quad \tilde{C}_2 = [C_x \quad -C_{2,P}], \quad \tilde{D}_2 = -D_P.$$

In order to proceed we need a number of definitions:

Definition 4.1. *Let $\Sigma = (A, B, C, D)$. By $\mathcal{T}_g(\Sigma)$ we denote the smallest subspace $\mathcal{T}$ of $\Re^n$ for which there exists a linear mapping K such that the following conditions are satisfied:*

$$(A - KC)\mathcal{T} \subseteq \mathcal{T}, \quad \mathrm{Im}\,(E - KD) \subseteq \mathcal{T}$$

and such that $A - KC_1 \,|\, \Re^n/\mathcal{T}$ is asymptotically stable. Similarly, by $\mathcal{V}_g(\Sigma)$ we denote the largest subspace $\mathcal{V}$ for which there exists a mapping $\tilde{F}$ such that the following conditions are satisfied:

$$(A - BF)\mathcal{V} \subseteq \mathcal{V}, \quad (C - DF)\mathcal{V} = \{0\}$$

and such that $A - BF \,|\, \mathcal{V}$ is asymptotically stable. □

Definition 4.2. *Let $\mathbf{X}_e$ denote the set of systems X satsfying the conditions of lemma 4.1 such that the corresponding auxiliary system Σ_a satisfies the following conditions:*

1. *$\mathcal{T}_g(\tilde{\Sigma}_{di}) \subseteq \mathcal{V}_g(\tilde{\Sigma}_{ci})$,*

2. *There exists a matrix* $\tilde{N}$ *such that*

$$\left(\begin{bmatrix}\tilde{A} & \tilde{E}\\ \tilde{C}_2 & D_x\end{bmatrix} + \begin{bmatrix}\tilde{B}\\ \tilde{D}_2\end{bmatrix}\tilde{N}\,[\tilde{C}_1 \quad \tilde{D}_1]\right)(\mathcal{T}_g(\tilde{\Sigma}_{di}) \oplus \Re^q) \subseteq (\mathcal{V}_g(\tilde{\Sigma}_{ci}) \oplus \{0\}), \tag{4.3}$$

where $\tilde{\Sigma}_{ci} = (\tilde{A}, \tilde{B}, \tilde{C}_2, \tilde{D}_2)$ *and* $\tilde{\Sigma}_{di} = (\tilde{A}, \tilde{E}, \tilde{C}_1, \tilde{D}_1)$. □

Next we note that since the matrix A_x is asymptotically stable and because of the properties of $\Sigma_{P,Q}$ as given in the previous section, it is simple to verify that

1. $(\tilde{A}, \tilde{B}, \tilde{C}_2, \tilde{D}_2)$ is right invertible and minimum phase, and
2. $(\tilde{A}, \tilde{E}, \tilde{C}_1, \tilde{D}_1)$ is left invertible and minimum phase.

This immediately implies that $(\tilde{A}, \tilde{B})$ is stabilizable and $(\tilde{A}, \tilde{C}_1)$ is detectable. Moreover, these conditions combined with the conditions 1 and 2 given in definition 4.2 guarantee (see [5]) that for the system Σ_a the disturbance decoupling with measurement feedback and internal stability is solvable. In other words, there exists a compensator Σ_F of the form (2.2) such that the interconnection $\Sigma_a \times \Sigma_F$ is internally stable and its transfer matrix is equal to 0.

Therefore, if we define the set

$$\mathbf{T}_e := \{\, T_{z_c w_c} \mid X \in \mathbf{X}_e \,\}, \tag{4.4}$$

then we have the following result.

Theorem 4.1. *The set* $\mathbf{T}$ *defined by (2.4) and the set* $\mathbf{T}_e$ *defined by (4.4) are equal, i.e.,* $\mathbf{T} = \mathbf{T}_e$. □

Proof : $(\Rightarrow)$: Let $T_{zw}(s) \in \mathbf{T}$. Hence by definition, there exists a controller $\Sigma_F \in \mathrm{C}$, which makes the closed-loop system on the left of (3.3) internally stable. It then follows from theorem 3.2 that such a controller makes the closed-loop system inside the dashed box on the right of (3.3) internally stable and yields $\|T_{z_{P,Q} w_{P,Q}}\|_\infty < 1$. Next, define X to be equal to the dashed box of (3.3). It is trivial to see that $\|X\|_\infty = \|T_{z_{P,Q} w_{P,Q}}\|_\infty < 1$ and that this system Σ_F solves the disturbance decoupling problem with measurement feedback and internal stability for the related auxiliary system Σ_a. It then follows from [5] that X must be such that the corresponding auxiliary system Σ_a satisfies the two conditions in definition 4.2. Hence, $X \in \mathbf{X}_e$ and $T_{z_C w_C} = T_{zw} \in \mathbf{T}_e$.

$(\Leftarrow)$: Conversely, assume that $T_{z_C w_C} \in \mathbf{T}_e$. By definition, there exists a system X of the form (4.1) such that $\|X\|_\infty < 1$ and the conditions in definition 4.2 hold. Hence, by [5], the disturbance decoupling problem with measurement feedback and internal stability is solvable for the corresponding auxiliary system Σ_a. Hence, there exists a stabilizing controller Σ_F such that the resulting system inside the dashed box of (3.3) is equal to X. By theorem 3.2, we have $\Sigma_F \in \mathrm{C}$. Moreover, the corresponding $T_{zw} = T_{z_C w_C} \in \mathbf{T}$. ∎

5. 'ALMOST' CHARACTERIZATION

It turns out that it is easier to define a bigger set $\mathbf{T}_a$ which contains set $\mathbf{T}$ and transfer matrices that are not in $\mathbf{T}$ are arbitraryly close to the set $\mathbf{T}$. To be more precise, for each element of $\mathbf{T}_a$ and for any positive scalar, say ε, there exists an element of $\mathbf{T}$ such that the difference between these two transfer matrices has H_∞-norm less than ε. Next, we will give a precise definition of the set $\mathbf{T}_a$:

Definition 5.1. *Let $\mathbf{X}_a$ denote the set of systems X of the form (4.1) where A_x is asymptotically stable, $\|X(s)\|_\infty < 1$ and $D_x = D_2 N D_1$ for some constant matrix N.*

Moreover, define the set $\mathbf{T}_a$ by:

$$\mathbf{T}_a := \{\, T_{z_C w_C} \mid X \in \mathbf{X}_a\}. \tag{5.1}$$

Now we can derive the following theorem.

Theorem 5.1. *The set $\mathbf{T}$ of (2.4) and the set $\mathbf{T}_a$ of (5.1) has the following relationship:*

$$\mathbf{T} \subseteq \mathbf{T}_a \subseteq \overline{\mathbf{T}}. \tag{5.2}$$

where the closure of the set $\mathbf{T}$ is with respect to the topology induced by the H_∞ norm. In other words, for any $\varepsilon > 0$ and for any $T_{z_C w_C} \in \mathbf{T}_a$, there exists a $T_{zw} \in \mathbf{T}$ such that $\|T_{zw} - T_{z_C w_C}\|_\infty < \varepsilon$. □

Proof : Part 1 It is trivial to verify that $\mathbf{X}_e \subseteq \mathbf{X}_a$. Hence, by definition, $\mathbf{T} = \mathbf{T}_e \subseteq \mathbf{T}_a$.

Part (2) For any $T_{z_C w_C} \in \mathbf{T}_a$, again by definition, there exists a system X of the form (4.1) with A_x stable, $\|X\|_\infty < 1$ and, moreover, there exists a matrix N such that $D_x = D_2 N D_1$. Since the range of D_2 and D_P are equal and since the kernel of $D_{P,Q}$ and D_1 are equal there exists a matrix $\tilde{N}$ such that

$$D_x = D_2 N D_1 = D_P \tilde{N} D_{P,Q} = \tilde{D}_2(-\tilde{N})\tilde{D}_1. \tag{5.3}$$

Finally, recall the following properties:

1. $(\tilde{A}, \tilde{B}, \tilde{C}_2, \tilde{D}_2)$ is right invertible and of minimum phase, and
2. $(\tilde{A}, \tilde{E}, \tilde{C}_1, \tilde{D}_1)$ is left invertible and of minimum phase.

It follows from [2,3] that the H_∞-almost disturbance decoupling problem with internal stability for the corresponding auxiliary system Σ_a is solvable. Hence, there exists Σ_F such that the corresponding closed-loop system inside the dashed box of (3.3), which we denote by X_F, is internally stable and is arbitrarily close to X in H_∞-norm. Let T_{zw} and $T_{z_C w_C}(s)$ be the closed-loop transfer matrices of the systems on the left and right respectively in (3.3). It is straightforward since Σ_C is stable that by making the difference

$X_F - X$ small enough that $T_{zw}(s)$ is also arbitrarily close to $T_{z_C w_C}(s)$ in H_∞-norm. More specifically, for any $\varepsilon > 0$, there exists Σ_F such that the corresponding T_{zw} satisfies $\|T_{zw} - T_{z_C w_C}\|_\infty < \varepsilon$. Furthermore, $\|T_{zw}\|_\infty < 1$ and hence $T_{zw} \in \mathbf{T}$. This completes the proof of the theorem. ∎

6. CONCLUSION

In this paper we have given a characterization of all stable closed loop systems we can obtain via a suitable feedback. The closed loop systems are parametrized via a stable system X with H_∞ norm less than 1. An exact characterization requires an extra constraint on X which is relatively difficult. However, if we are satisfied with an approximate characterization then the system X has to satisfy only one extra constraint which is very simple.

No explicit characterization of all suitable controllers is given. It is our belief that a simple characterization of all controllers as given in [1,6] can not be given in the singular case. This still remains an interesting open problem.

Construction of suitable controllers in the singular case can be done via solving a almost disturbance decoupling problem. Algorithms can e.g. be found in [2,3].

References

[1] J. Doyle, K. Glover, P.P. Khargonekar, B.A. Francis, “State space solutions to standard H_2 and H_∞ control problems”, *IEEE Trans. Aut. Contr.*, Vol. 34, No. 8, 1989, pp. 831-847.

[2] H.K. Ozcetin, A. Saberi, P. Sannuti, “Design for almost disturbance decoupling problem with internal stability via state or measurement feedback– singular perturbation approach”, To appear in *Int. J. Contr.*.

[3] H.K. Ozcetin, A. Saberi and Y. Shamash, “An H_∞ almost disturbance decoupling technique for nonstrictly proper systems”, Submitted to CDC Brighton, U.K., 1991.

[4] A.A. Stoorvogel, “The singular H_∞ control problem with dynamic measurement feedback”, *SIAM J. Contr. & Opt.*. Vol. 29, No. 1, 1991, pp. 160-184.

[5] A.A. Stoorvogel, J.W. van der Woude, “The disturbance decoupling problem with measurement feedback and stability for systems with direct feedthrough matrices”, To appear in *Syst. & Contr. Letters*,

[6] G. Tadmor, “Worst-case design in the time domain: the maximum principle and the standard H_∞ problem”, *Math. Contr. Sign. & Syst.*, Vol. 3, 1990, pp. 301-324.

OUTPUT FEEDBACK STABILIZING CONTROL WITH BOUNDS ON DISTURBANCE ATTENUATION

Brian C. Fabien
Assistant Professor
Department of Mechanical Engineering
Ohio University, Athens Ohio 45701

ABSTRACT

This paper considers the design of direct output static and dynamic stabilizing controllers for linear systems. Here, controllers that stabilize the closed-loop system are required to satisfy an H_∞ disturbance attenuation constraint. Sufficient conditions for the existence of such controllers are presented in the form of two coupled nonlinear matrix equations. These conditions are derived by solving a mixed-norm H_2/H_∞ optimal control problem. Solution of these equations lead to a fixed-order stabilizing controller gain matrix that minimizes the H_∞ norm of the closed-loop transfer function. As an example the controller synthesis technique is applied to an undamped single degree-of-freedom oscillator.

1. INTRODUCTION

This paper is concerned with the synthesis of static and dynamic output feedback controllers for Linear Time Invariant (LTI) systems. The controllers are such that the resultant closed-loop system is stable and satisfies an H_∞ disturbance attenuation constraint. The class of LTI systems considered are those for which the measured output is noise-free or is affected by a colored noise process. Of particular interest are linear systems that are open-loop unstable (or marginally stable), such as, undamped structures, electromagnetic levitation devices, aircraft, etc.

Kabamba et al. (1987) examines the problem of finding static and dynamic controllers for LTI systems, presenting, necessary and sufficient conditions for the existence of stabilizing controllers in the form of a set of nonlinear matrix equations, along with a homotopy procedure for solving these equations. However, the disturbance attenuation properties of the closed-loop system are not addressed.

Recently several authors have considered the problem of designing controllers that result in stable closed-loop systems that satisfy an H_∞ disturbance attenuation bound. Specifically, Petersen (1987) describes a method for designing H_∞ constrained state feedback controllers. The method is based on finding positive definite solutions to a specific Riccati equation. Khargonekar et al. (1988) uses the same Riccati equation to study static and dynamic state feedback controllers, where it is shown that the degree of disturbance attenuation that can be achieved by dynamic state feedback is no better than that which can be achieved by static state feedback.

Minimizing an LQG cost function with an H_∞ disturbance attenuation constraint, Bernstein and Haddad (1989, 1990) present design methods for full- and reduced-order dynamic feedback controllers. Also, Steinbuch and Bosgra (1991) solve a similar mixed-norm H_2/H_∞ optimal control problem for static and fixed order dynamic controllers.

The purpose of this paper is not to solve an optimal control problem per se. However, the mathematical techniques of optimal control theory are used to obtain the main result of the paper. Here, the sufficient conditions for the existence of stabilizing static and dynamic output feedback controllers that satisfy an H_∞ disturbance attenuation bound are presented. These conditions take the form of a set of nonlinear matrix equations and are obtained by solving a mixed-norm H_2/H_∞ optimal control problem.

The content of the of the paper can be summarized as follows: In section 2 the controller synthesis problem is formulated. Section 3 presents the sufficient conditions for the existence of controllers that will solve the problem stated in section 2. Section 4 gives an example problem and the paper concludes in section 5 with some observations.

2. PROBLEM FORMULATION

Consider the LTI system

$$\begin{aligned} \dot{x}_s &= A_s x_s + B_s u + w_s \\ y &= C_s x_s \\ z &= E_s x_s \end{aligned} \tag{1}$$

where $x_s \in R^{n_p}$ denotes the plant state vector, $u \in R^{m_p}$ the input vector, $y \in R^{p_p}$ the measured output, $z \in R^q$ the controlled output, w_s the input disturbance vector. The real matrices A_s, B_s, C_s and E_s are of appropriate dimensions. Furthermore, B_s and C_s are of full rank.

The controller for the system is assumed to have the form

$$\begin{aligned} \dot{x}_c &= A_c x_c + B_c y + w_c \\ u &= C_c x_c + D_c y \end{aligned} \tag{2}$$

where $x_c \in R^{n_c}$ denotes the controller state vector, and A_c, B_c, C_c, and D_c are appropriately dimensioned matrices. The controller state vector is assumed to be corrupted by a noise signal, w_c, that is a white, gaussian, zero mean, random process with independent initial conditions. Also, w_c is uncorrelated with the inputs w_s. The dimension of the controller, n_c, is such that $0 \le n_c \le n_p$. When $n_c = 0$ the controller is a direct output feedback controller, whereas $n_c = n_p$ implies full-order dynamic compensation.

Using the control law (2) in (1) the resultant closed-loop system can be written as

$$\begin{aligned} \dot{x} &= \tilde{A}x + w \\ z &= Ex \end{aligned} \tag{3}$$

where $x = \begin{bmatrix} x_s \\ x_c \end{bmatrix} \in R^n$, $w = \begin{bmatrix} w_s \\ w_c \end{bmatrix}$, $\tilde{A} = A + BFC$, $A = \begin{bmatrix} A_s & 0 \\ 0 & 0 \end{bmatrix} \in R^{n \times n}$, $B = \begin{bmatrix} B_s & 0 \\ 0 & I \end{bmatrix} \in R^{n \times m}$,

$C = \begin{bmatrix} C_s & 0 \\ 0 & I \end{bmatrix} \in R^{p \times n}$, $E = \begin{bmatrix} E_s & 0 \end{bmatrix} \in R^{q \times n}$, $F = \begin{bmatrix} D_c & C_c \\ B_c & A_c \end{bmatrix} \in R^{m \times p}$, $n = n_s + n_c$, $m = m_p + n_c$, $p = p_p + n_c$ and I denotes the appropriately dimensioned identity matrix. The closed-loop transfer function from the input disturbance, w, to the controlled output, z, is given by

$$H(s) = E(sI-\tilde{A})^{-1} \tag{4}$$

With the system defined by equations (1) thru (4) the problem to be solved can be stated as follows:
Problem: *Stabilization with disturbance attenuation constraint* (SDAC).

(i) Find conditions for the existence of a finite controller gain matrix, F, such that Re $\lambda(\tilde{A}) < 0$ and $\|H(s)\|_\infty \leq \gamma > 0$.

(ii) If such a controller exists compute it.

(iii) Compute the minimum value of γ for which stabilizing controllers, F, exist. In other words find a fixed-order controller, F, that minimizes the H_∞ norm of the closed-loop transfer function H(s).

The H_∞ norm of the the transfer function H(s) is defined by

$$\|H(s)\|_\infty = \max_{\omega \in R} \bar{\sigma}(H(i\omega))$$

with $\bar{\sigma}$ being the maximum singular value.

The first condition indicates the desire to have a stable closed-loop system with a prescribed degree of disturbance attenuation, γ, from the input disturbance to the controlled output. The second and third conditions express the desire to have a computational schemes for finding these stabilizing controllers as well controllers that will provide the best possible disturbance attenuation characteristics.

The solution of the SDAC problem has some obvious practical significance. First, it is noted that stability is a fundamental requirement for any closed-loop control system. However, control systems must also satisfy other performance requirements such as speed of response, accuracy (tracking and steady-state), insensitivity to parameter variation, disturbance and noise rejection, minimum control effort, etc. In general it is difficult to design controllers that meet all of these conditions simultaneously and some compromise in the systems performance must be made. Nonetheless, it can be shown that minimizing the H_∞ norm of the transfer function H(s) provides robustness of the closed-loop system to structured uncertainties in the plant (Maciejowski, 1989). Thus solution of the SDAC problem provide an approach to meeting some of the general requirements of control systems.

3. STABILIZING CONTROL WITH DISTURBANCE ATTENUATION

3.1 Stabilizing Control

En route to solving the SDAC problem, consider the problem of finding a controller, F, that will stabilize the system (3). Here the disturbance attenuation constraint is not necessarily satisfied. Sufficient conditions for the existence of such a controller is given in the following result.

Theorem 1: The system (3) is stabilizable by a finite feedback control, F, if there exists symmetric positive definite matrices K and S that solve

$$A^TK + KA - KBB^TKSC^TC - C^TCSKBB^TK + Q = 0 \tag{5a}$$
$$AS + SA^T - BB^TKSC^TCS - SC^TCSKBB^T + I = 0 \tag{5b}$$

where $Q \in R^{n \times n}$ is a symmetric positive definite matrix. In such case

$$F = -B^TKSC^T \tag{5c}$$

Proof. Substitute (5c) into (5a) and (5b) to get

$$\tilde{A}^TK + K\tilde{A} + Q = 0 \tag{6a}$$
$$\tilde{A}S + S\tilde{A}^T + I = 0 \tag{6b}$$

Thus, if K and S are positive definite matrices the Lyapunov equations above imply that Re $\lambda(\tilde{A}) < 0$ and F is a stabilizing controller. □

Remark 3.1: This result can be obtained by finding an extremum of the scalar cost function, $J_0 = \mathrm{tr}(F^TF) + \lim_{t\to\infty} \mathbb{E}(x^TQx)$ where $\mathrm{tr}(\cdot)$ denotes the trace, $\mathbb{E}(\cdot)$ the expected value. If it is assumed that $\mathrm{Re}\ \lambda(\tilde{A}) < 0$ and $\mathbb{E}(w(t)w(\tau)^T) = I\delta(t-\tau)$ then the cost function can be written as $J_0 = \mathrm{tr}(F^TF + SQ)$ where the steady-state closed-loop state covariance, S, satisfies the Lyapunov equation (6b). (Note that the cost function will have a finite value only if $\mathrm{Re}\ \lambda(\tilde{A}) < 0$). Adjoin the constraint equation (6b) to the cost function J_0 via the method of Lagrange multipliers to get the augmented cost function, $L_0 = \mathrm{tr}(F^TF + SQ + K(\tilde{A}S + S\tilde{A}^T + I))$. Here, K is the matrix of Lagrange multipliers. The first order necessary conditions for J_0 to be extremum, i.e., $\frac{\partial L_0}{\partial S} = \frac{\partial L_0}{\partial K} = \frac{\partial L_0}{\partial F} = 0$ produces equations (5a), (5b) and (5c).

Controllers, F, that satisfy equations (5a,b,c) only guarantee that the cost function J_0 is extremum, i.e., the cost function is not necessarily a global minimum (thus, F is not necessarily optimal). However, the solution to this optimization problem provides the methodology used to obtain a solution to SDAC problem.

It should also be noted that (5a,b,c) are the same equations obtained by Kabamba et al. (1987) for the case where $w = 0$, $Q = I$ and $\mathbb{E}(x(0)x(0)^T) = I$.

3.2 Selection of the matrix Q

Henceforth it is assumed that the symmetric positive definite matrix, Q, is selected according to

$$Q = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{12}^T & Q_{22} \end{bmatrix} \tag{7}$$

where $Q_{11} \in \mathcal{R}^{n_s \times n_s}$, $Q_{12} \in \mathcal{R}^{n_s \times n_c}$, $Q_{22} \in \mathcal{R}^{n_c \times n_c}$. Also, (i) $Q_{11} > 0$, (ii) $Q_{22} > 0$, (iii) $Q_{11} - Q_{12} Q_{22}^{-1} Q_{12}^T > 0$ and (iv) $Q_{12}^T Q_{12} \neq 0$.

Conditions (i), (ii) and (iii) ensure that Q is positive definite. Condition (iv) ensures that the stabilizing controllers that satisfy (5a,b) are such that $B_c \neq 0$ and $C_c \neq 0$. To see this write F given by equation (5c) in partitioned form to get

$$F = \begin{bmatrix} D_c & C_c \\ B_c & A_c \end{bmatrix} = -\begin{bmatrix} B_s^T & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} K_{11} & K_{12} \\ K_{12}^T & K_{22} \end{bmatrix} \begin{bmatrix} S_{11} & S_{12} \\ S_{12}^T & S_{22} \end{bmatrix} \begin{bmatrix} C_s^T & 0 \\ 0 & I \end{bmatrix}$$

where K and S are partitioned in a manner similar to Q in equation (7). Thus,

$$B_c = -\left(K_{12}^T S_{11} + K_{22} S_{12}^T\right) C_s^T \text{ and } C_c = -B_s^T\left(K_{11}S_{12} + K_{12}S_{22}\right).$$

Now writing the Lyapunov equations (6a,b) in partitioned form yields the following unique relations for the off-diagonal terms:

$$K_{11}B_sC_c + \left(A_s^T + C_s^T D_c B_s^T\right)K_{12} + K_{12}A_c + C_s^T B_c^T K_{22} = -Q_{12}$$

$$S_{11}C_s^T B_c^T + \left(A_s + B_s D_c C_s\right)S_{12} + S_{12}A_c^T + B_s C_c S_{22} = 0$$

Suppose $Q_{12} = 0$, then using the relations for B_c and C_c it can be seen that positive definite matrices K and S that satisfy these off-diagonal terms are of the form; $K_{11} > 0$, $K_{22} > 0$, $S_{11} > 0$, $S_{22} > 0$, $K_{12} = S_{12} = 0$. However, this implies that $B_c = C_c = 0$; in which case the dynamic compensator has no influence on the control signal, **u**. Thus to ensure that $B_c \neq 0$ and $C_c \neq 0$ the restriction (iv) in equation (7) is imposed. In terms of the optimization problem posed to obtain equations (5a,b,c), the restriction (iv) indicates that there must be cross-weighting between the states of the plant and controller.

3.3 Disturbance Attenuation

Although the results of Theorem 1 can be used to obtain stabilizing controllers it does not ensure disturbance attenuation as posed in the SDAC problem. To satisfy this disturbance attenuation constraint consider the following Lemma.

Lemma 1: For the linear system defined by equations (3) and (4), if there exists a stabilizing control, F, such that the Riccati equation

$$\tilde{A}\Lambda + \Lambda\tilde{A}^T + \frac{1}{\gamma^2}\Lambda E^T E\Lambda + I = 0 \tag{8}$$

admits a positive definite solution for Λ, then $\|H(s)\|_\infty \leq \gamma > 0$.

Proof: See Petersen (1987). □

Thus, to solve the SDAC problem the constraint (8) given in Lemma 1 must somehow be incorporated into the results of Theorem 1. To accomplish this the mixed-norm H_2/H_∞ approach of Bernstein and Haddad (1989) is used. Specifically, consider the following problem: Find the stabilizing controller, F, that minimizes the cost function

$$J_1 = \mathrm{tr}(F^T F + \Lambda Q) \tag{9}$$

$$\text{subject to } \tilde{A}\Lambda + \Lambda\tilde{A}^T + \frac{1}{\gamma^2}\Lambda E^T E\Lambda + I = 0, \quad \Lambda > 0.$$

This problem is similar to the one posed in Theorem 1, however, the constraint equation (6b) is replaced by equation (8) which enforces the required disturbance attenuation constraint. Solving the problem defined by (9), using the method of Lagrange multipliers as in Theorem 1, yields the the solution to the SDAC problem:

Theorem 2: The system (3) is stabilizable by a finite feedback control, F, and the closed-loop system satisfies the disturbance attenuation bound, $\|H(s)\|_\infty \leq \gamma > 0$, if there exist symmetric positive definite matrices K and Λ that solve

$$A^T K + KA - KBB^T K\Lambda C^T C - C^T C\Lambda KBB^T K + \frac{1}{\gamma^2}\left(K\Lambda E^T E + E^T E\Lambda K\right) + Q = 0 \tag{10a}$$

$$A\Lambda + \Lambda A^T - BB^T K\Lambda C^T C\Lambda - \Lambda C^T C\Lambda KBB^T + \frac{1}{\gamma^2}\Lambda E^T E\Lambda + I = 0 \tag{10b}$$

In such case $F = -B^T K\Lambda C^T$ (10c)

Proof: *Stability*. Substitute (10c) into (10a) to get

$$\tilde{A}^T K + K\tilde{A} = -\left[Q + \frac{1}{\gamma^2}K\Lambda E^T E + \frac{1}{\gamma^2}E^T E\Lambda K\right]$$

Thus, if K and Λ are positive definite solutions to (10a) and (10b) the Lyapunov equation given above implies that $\mathrm{Re}\ \lambda(\tilde{A}) < 0$.

Disturbance Attenuation. Substitute (10c) into (10b) to get

$$\tilde{A}\Lambda + \Lambda\tilde{A}^T + \frac{1}{\gamma^2}\Lambda E^T E\Lambda + I = 0$$

Applying Lemma 1 yields the desired result. □

Remark 3.2: If there exists a controller, F, that satisfies (10a) and (10b) then it also satisfies (5a) and (5b). Furthermore, subtracting the cost functions gives, $J_1 - J_0 = \mathrm{tr}(\Lambda - S)Q$. Now, subtracting (6b) from (8) yields,

$$\tilde{A}(\Lambda - S) + (\Lambda - S)\tilde{A}^T = -\frac{1}{\gamma^2}\Lambda E^T E\Lambda$$

If Theorems 1 and 2 hold then the equation given above indicates that $(\Lambda - S)$ is positive semi-definite, in other words $J_1 - J_0 \geq 0$. Therefore, the cost function J_1 is an upper bound of the cost function J_0. (A similar result was noted by Bernstein and Haddad (1989) in the solution of the mixed-norm H_2/H_∞ standard problem).

Remark 3.3: Equations (10a) and (10b) can be rewritten in terms of $\tilde{A}$ as

$$\left(\tilde{A}^T + \frac{1}{\gamma^2}E^T E\Lambda\right)K + K\left(\tilde{A} + \frac{1}{\gamma^2}\Lambda E^T E\right) + Q = 0 \qquad (11a)$$

$$\tilde{A}\Lambda + \Lambda\tilde{A}^T + \frac{1}{\gamma^2}\Lambda E^T E\Lambda + I = 0 \qquad (11b)$$

respectively. These equations can be viewed as a modified Lyapunov equation, (11a), coupled with a Riccati equation, (11b). If the disturbance attenuation constraint is relaxed, i.e., $\gamma \to \infty$, then (11a) and (11b) reduce to the two coupled nonlinear matrix equations given by (6a) and (6b) respectively, and $J_1 = J_0$.

From the results of this section it can be seen that it is possible to improve the disturbance attenuating properties of the controller given in section 3.1. However, this is done at a cost larger than J_0. In general this implies higher controller gains are required to improve the disturbance attenuation characteristics of the closed-loop system.

3.4 Some Generalizations

The system described by equation (3) requires that each state derivative be affected by an input disturbance. This assumption can be relaxed by considering an the input disturbance of the form $w = D\bar{w}$ with $\Xi(\bar{w}(t)\bar{w}(\tau)^T) = I\delta(t-\tau)$. (It is assumed that appropriate choices have been made for the dimensions of the matrix D and the vector $\bar{w}$). In this case equations (3), (4), (5b), (8), and (10b) are modified as follows:

$$(3) \rightarrow \quad \dot{x} = \tilde{A}x + D\bar{w} \qquad (3')$$

$$(4) \rightarrow \quad H(s) = E(sI-\tilde{A})^{-1}D \qquad (4')$$

$$(5b) \rightarrow \quad AS + SA^T - BB^T KSC^T CS - SC^T CSKBB^T + DD^T = 0 \qquad (5b')$$

$$(8) \rightarrow \quad \tilde{A}\Lambda + \Lambda\tilde{A}^T + \frac{1}{\gamma^2}\Lambda E^T E\Lambda + DD^T = 0 \qquad (8')$$

$$(10b) \rightarrow \quad A\Lambda + \Lambda A^T - BB^T K\Lambda C^T C\Lambda - \Lambda C^T C\Lambda KBBT + \frac{1}{\gamma^2}\Lambda E^T E\Lambda + DD^T = 0 \qquad (10b')$$

If it is further assumed that the pair $(\tilde{A},D)$ is controllable then Theorem 1, Lemma 1 and Theorem 2 still hold with the substitutions given above. This generalization puts an additional constraint on the allowable controller gain matrices. Specifically, F is restricted to the set of stabilizing controller gains for which the controllability of $(\tilde{A},D)$ is satisfied.

Note also that the results obtained can easily accommodate the case of state feedback, i.e., $C = I$.

It is also possible to consider alternate cost functions (e.g. Levine et al. (1971), Steinbuch and Bosgra (1991)) but as far as the problem of stabilization with disturbance attenuation is concerned these cost functions do not provide equations that are any less difficult to solve.

4. EXAMPLE PROBLEM

4.1 Example

Consider a single degree-of-freedom undamped oscillator, with dynamic equations given by

$$\dot{x}_s = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} x_s + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} w_s$$

$$y = \begin{bmatrix} 0 & 1 \end{bmatrix} x_s; \quad z = x_s$$

Note that the system is marginally stable having poles at ±i.

The following controller designs are considered:

Case (1) Full-state feedback control, $C = I$, $u = Fx_s$, $Q = \mathrm{diag}(1,0.001)$.

Case (2) Direct output feedback control, $n_c = 0$, $u = Fy$, $Q = \mathrm{diag}(1,0.001)$.

Case (3) A first order dynamic controller, i.e., $n_c = 1$, $\dot{x}_c = a_c x_c + b_c y + w_c$,

$$u = c_c x_c + d_c y, \; Q = \frac{1}{2}\begin{bmatrix} 2 & 0 & 0 \\ 0 & 0.002 & 1 \\ 0 & 1 & 2 \end{bmatrix}.$$

Recall that the disturbance signals, w_s and w_c, are white, gaussian, zero mean, uncorrelated, random processes with independent initial conditions and unit intensity. Also, for the SDAC problem the matrix, Q, is selected according to the conditions given in equation (9). In general, Q can be used as a set of design parameters to adjust the structure and magnitude of the controller gains. Here, however an arbitrary selection is made.

Table 1. Stabilizing Controllers: Solution of equation (5a,b,c), Example 1.

Case	Gain, F	Cost, J_0	$\|\|H(s)\|\|_\infty$	Closed-loop Poles
1	$[-0.310 \;\; -0.703]$	2.113	3.019	-0.3512 ± 1.0892i
2	-0.719	2.267	2.981	-0.3593 ± 0.9332i
3	$\begin{bmatrix} -0.701 & 0.015 \\ -0.180 & -0.639 \end{bmatrix}$	3.393	3.035	-0.3513 ± 0.9378i -0.6373

Table 1 presents the results obtained by applying a continuation algorithm to solve equations (5a,b,c). (The details of this solution technique is given in a forthcoming paper, Fabien (1991)). These controllers satisfy equations (5a,b). The optimal control cost function, J_0, is least for full-state feedback control and most for the dynamic compensator in case (3). It is interesting to note that in all cases the actual H_∞ norm of the closed-loop transfer function, $||H(s)||_\infty$, is approximately equal to 3.0 and the closed-loop poles are approximately in the same location.

Table 2 presents the results obtained by applying the continuation algorithm to solve equation (10a,b,c). The algorithm essentially finds solutions to (10a,b,c) along a space curve where γ is considered as the continuation parameter. By varying γ in an incremental fashion solutions to (10a,b,c) are found in the range $\infty > \gamma \geq \gamma_{min}$. Here, γ_{min} is the best achievable disturbance attenuation for a controller of a given order. Furthermore, γ_{min} is a turning point on the solution space curve.

The results in Table 2 represent solutions to the SDAC problem. In all cases the H_∞ norm of the closed-loop transfer function has been reduced when compared to the stabilizing control obtained in Table 1. However, the cost function J_1 is larger than J_0 in all cases. Note also the $\gamma_{min} = ||H(s)||_\infty$ in all cases, indicating that the disturbance attenuation constraint equation is satisfied identically. In general, the improvement in disturbance attenuation (when compared to the results of Table 1) has been achieved by increasing the magnitude of the controller gains.

Overall, the state feedback controller, case (1), provides the best possible disturbance attenuation. Recall that Khargonekar et al. (1988) have shown that the level of disturbance attenuation, γ, provided by dynamic state feedback is no better than that which can be achieved by static state feedback. From the results presented in Table 2 it can be seen that in the case of output feedback there exists dynamic compensators that provide better disturbance attenuation that static output feedback.

Table 2. Stabilizing Control: Solution of equation (10a,b,c), Example 1.

Case	Gain, F	Cost J_1	γ_{min}	$\|\|H(s)\|\|_\infty$	Closed-loop Poles
1	$[-8.504\ -4.338]$	91.635	1.105	1.105	-2.1688 ± 2.1910i
2	-1.391	4.682	2.0	2.0	-0.6955 ± 0.7185i
3	$\begin{bmatrix} 0.742 & 1.366 \\ -5.260 & -3.72 \end{bmatrix}$	56.363	1.786	1.786	-0.9330 ± 1.5707i -1.1158

The dynamic compensator of case (3) yields an improvement in the closed-loop disturbance attenuation when compared to the direct output feedback controller. However, this disturbance attenuation level is not as low as that which can be achieved using state feedback control.

Fig. 1a and 1b illustrate the typical frequency response characteristics for the closed-loop systems. In Fig. 1a plots of $\left|\frac{x_{s1}}{w_{s1}}\right|$ vs. frequency, ω, for the controllers in Table 1 is presented. All the controllers exhibit approximately the same behavior. Fig. 1b shows plots of $\left|\frac{x_{s1}}{w_{s1}}\right|$ vs. ω, for the controllers in Table 2. At frequencies $\omega < 2$ the state feedback controller provides better disturbance attenuation than the other controllers. Also, the dynamic compensator, case (3), provides better disturbance attenuation than the static output feedback controller.

5. CONCLUSIONS

In this paper sufficient conditions for the existence of stabilizing output feedback controllers that satisfy a disturbance attenuation constraint are presented. These conditions take the form of a pair of coupled nonlinear matrix equations and are obtained by solving a mixed-norm H_2/H_∞ optimal control problem.

From the example problem considered it has been observed that the technique presented in this paper is quite effective at finding stabilizing static and dynamic output feedback controllers that achieve some disturbance attenuation $\gamma > \gamma_{min}$, where γ_{min} is the best achievable disturbance attenuation for a controller of a given order. Numerical experience has shown that it is possible to find a controller, F, for which γ is arbitrarily close to γ_{min}, however, this must be done at great computational expense.

6 REFERENCES

Bernstein, D. and Haddad, W., "LQG Control with an H_∞ Performance Bound: a Riccati Equation Approach," *IEEE Trans. AC*, Vol. 34, 1989, pp. 293-305.

Fabien, B. C., "Static and Dynamic Stabilizing Control with an H_∞ Bound on Disturbance Attenuation," *to appear*.

Haddad, W. and Bernstein, D., "Generalized Riccati Equations for the Full- and Reduced-Order Mixed-Norm H_2/H_∞ Standard Problem," *Systems and Control Letters*, Vol. 14, pp. 185-197, 1990.

Kabamba, P. T. and Longman, R. W., "An Integrated Approach to Reduced-Order Control Theory," *Optimal Control Applications and Methods*, Vol. 4, pp. 405-415, 1983.

Kabamba, P. T., Longman, R. W. and Jian-Guo, S., "A Homotopy Approach to Feedback Stabilization of Linear Systems," *J. Guidance, Control and Dynamics*, Vol. 10, No. 5, pp. 422-432, 1987.

Levine, W. S., Johnson, T. L. and Athens, M., "Optimal Limited State Variable Feedback Controllers for Linear Systems," *IEEE Trans. AC*, Vol. 16, pp. 785-792, 1971.

Maciejowski, J. M., *Multivariable Feedback Design*, Addison-Wesley, 1989.

Petersen, I., "Disturbance Attenuation and H_∞ Optimization: A Design Method Based on the Algebraic Riccati Equation," *IEEE Trans. AC*, Vol. 32, pp. 427-429, 1987.

Steinbuch, M. and Bosgra, O., "Necessary Conditions for Static and Fixed Order Dynamic Mixed H_2/H_∞ Optimal Control," *Proceedings 1991 American Control Conference*, pp. 1137-1142, 1991.

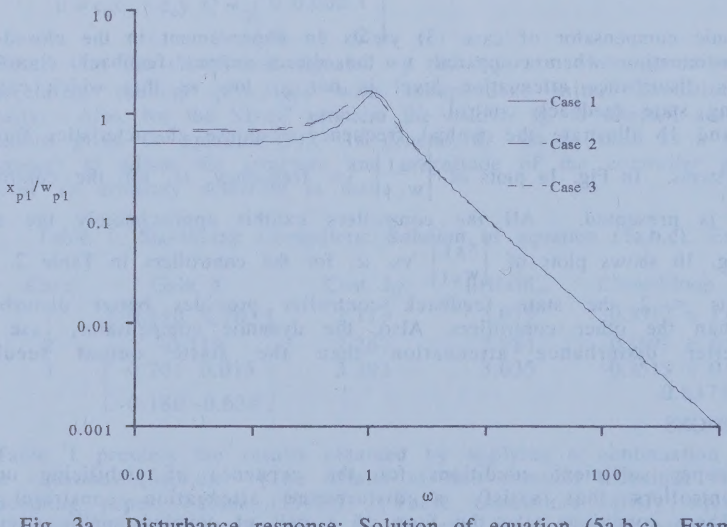


Fig. 3a Disturbance response: Solution of equation (5a,b,c), Example 1

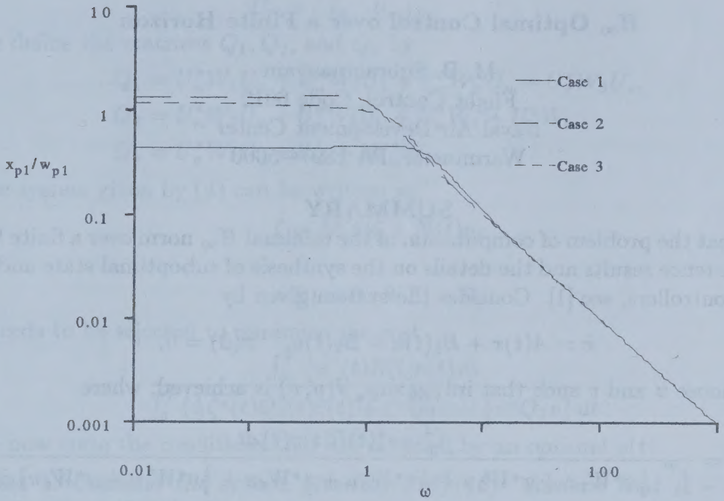


Fig. 1b Disturbance response: Solution of equation (10a,b,c), Example 1

H_∞ Optimal Control over a Finite Horizon

M. B. Subrahmanyam
Flight Control, Code 6012
Naval Air Development Center
Warminster, PA 18974-5000

SUMMARY

We treat the problem of computation of the minimal H_∞ norm over a finite horizon. For the existence results and the details on the synthesis of suboptimal state and output feedback controllers, see [1]. Consider the system given by

$$\dot{x} = A(t)x + B_1(t)u + B_2(t)v, \quad x(0) = 0, \tag{1}$$

We choose u and v such that $\inf_{v \neq 0} \sup_u J(u,v)$ is achieved, where

$$J(u,v) = \frac{\int_0^T \frac{1}{2} v^*(t)R(t)v(t)\,dt}{\int_0^T \{\frac{1}{2}x^*W_1x + x^*W_2u + \frac{1}{2}u^*W_3u + x^*W_4v + \frac{1}{2}v^*W_5v + u^*W_6v\}\,dt}. \tag{2}$$

We will use the adjoint variables associated with the maximization part of (2) as state variables for the minimization part of (2).

Let

$$\lambda = \inf_{v \neq 0} \sup_u J(u,v). \tag{3}$$

Let θ be the adjoint variable associated with the maximization problem. From the maximum principle, we have

$$\begin{pmatrix} \dot{x} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \hat{A} & \hat{B} \\ \hat{C} & -\hat{A}^* \end{pmatrix} \begin{pmatrix} x \\ \theta \end{pmatrix} + \begin{pmatrix} G_1 \\ G_2 \end{pmatrix} v, \tag{4}$$

with

$$x(0) = 0, \ \theta(T) = 0, \tag{5}$$

where

$$\begin{aligned} \hat{A} &= A - B_1W_3^{-1}W_2^*, \\ \hat{B} &= B_1W_3^{-1}B_1^*, \\ \hat{C} &= W_1 - W_2W_3^{-1}W_2^*, \\ G_1 &= B_2 - B_1W_3^{-1}W_6, \\ G_2 &= W_4 - W_2W_3^{-1}W_6. \end{aligned} \tag{6}$$

Let

$$\zeta = \begin{pmatrix} x \\ \theta \end{pmatrix}, \tag{7}$$

$$M = \begin{pmatrix} \hat{A} & \hat{B} \\ \hat{C} & -\hat{A}^* \end{pmatrix}, \tag{8}$$

$$N = \begin{pmatrix} G_1 \\ G_2 \end{pmatrix}, \tag{9}$$

$$U_z = W_3^{-1}\,(-W_2^* \quad B_1^*) \tag{10}$$

$$U_v = -W_3^{-1}W_6, \tag{11}$$

and

$$U_n = (\, I_n \quad 0_n \,). \tag{12}$$

We define the matrices Q_1, Q_2, and Q_3 by

$$Q_1 = U_n^* W_1 U_n + U_n^* W_2 U_z + U_z^* W_2^* U_n + U_z^* W_3 U_z, \tag{13}$$

$$Q_2 = U_n^* W_2 U_v + U_z^* W_3 U_v + U_n^* W_4 + U_z^* W_6, \tag{14}$$

$$Q_3 = U_v^* W_3 U_v + W_5 + 2U_v^* W_6. \tag{15}$$

The system given by (4) can be written as

$$\dot{\zeta} = M(t)\zeta + N(t)v, \tag{16}$$

with

$$x(0) = 0,\ \theta(T) = 0, \tag{17}$$

and v needs to be selected to minimize the cost

$$\frac{\int_0^T \frac{1}{2} v^*(t) R(t) v(t)\, dt}{\int_0^T \{\frac{1}{2}\zeta^*(t) Q_1(t) \zeta(t) + \zeta^* Q_2 v + \frac{1}{2} v^* Q_3 v\}\, dt}. \tag{18}$$

We now state the conditions that are satisfied by an optimal $v(t)$.

THEOREM 1. *Consider the system given by (16)-(18). Assume that $R - \lambda Q_3$ is invertible for all $t \in [0, T]$. If $v_0(t)$ minimizes (18), then there exists a nonzero $\rho(t) = (\, p^*(t) \quad q^*(t) \,)^*$ such that*

$$\frac{d\rho}{dt} = -M^* \rho - \lambda Q_1 \zeta - \lambda Q_2 v, \tag{19}$$

where $p(t)$ and $q(t)$ are components of the adjoint vector corresponding to $x(t)$ and $\theta(t)$ respectively, such that

$$\begin{aligned} x(0) = 0,\ \theta(T) = 0, \\ p(T) = 0,\ q(0) = 0, \end{aligned} \tag{20}$$

where

$$\lambda = \inf_{v \neq 0} \frac{\int_0^T \frac{1}{2} v^* R v\, dt}{\int_0^T \{\frac{1}{2}\zeta^* Q_1 \zeta + \zeta^* Q_2 v + \frac{1}{2} v^* Q_3 v\}\, dt}, \tag{21}$$

and

$$v_0(t) = (R - \lambda Q_3)^{-1} \{\lambda Q_2^* \zeta + N^* \rho\}. \tag{22}$$

We now give a criterion for the estimation of λ. Notice that $\lambda = \min_{v \neq 0} \max_u J(u, v)$ and gives a measure of performance of the optimal controller under worst-case conditions corresponding to $v_0(t)$. In the H_∞ case, the evaluation of λ would entail the γ-iteration.

THEOREM 2. *Let λ be the smallest positive value for which the boundary value problem given by (16),(19), (20), and (22) has a solution (ζ, ρ) with $\int_0^T \{\frac{1}{2}\zeta^* Q_1 \zeta + \zeta^* Q_2 v + \frac{1}{2} v^* Q_3 v\}\, dt > 0$, where $v \triangleq (R - \lambda Q_3)^{-1}\{\lambda Q_2^* \zeta + N^* \rho\}$. Then λ is the minimum value of (18), (ζ, ρ) is an optimal pair and $v = (R - \lambda Q_3)^{-1}\{\lambda Q_2^* \zeta + N^* \rho\}$ is the worst exogenous input.*

The criterion in Theorem 2 can be used to devise computational tools for the evaluation of the infimal H_∞ norm in the finite horizon case. Our computational experience shows that for time-invariant problems, the infimal H_∞ norm in the finite horizon case approaches that in the infinite horizon case as the final time T becomes large.

REFERENCE

[1] M. B. SUBRAHMANYAM, H_∞ Optimal Control Theory over a Finite Horizon, Technical Report, No. NADC-91077-60, Naval Air Development Center, Warminster, PA, September 1991.

Numerical Computation of Inner Factors for Distributed Parameter Systems [1]

David S. Flamm
Department of Electrical Engineering, Princeton University
Princeton, NJ 08544

1 Introduction

The motivation for the developments in this paper is interest in explicitly computing solutions to H^∞-optimal control problems for distributed parameter systems, such as a damped flexible beam [1]. In this case one has an irrational transfer function

$$P(s) = N(s)/D(s) \tag{1}$$

where N and D are both functions in H^∞. The solution to the mixed sensitivity problem [6],[2] involves finding the largest solution μ_0 on an interval $[a, b] \subset \mathbb{R}$ to an equation of the form

$$N_i(j\zeta(\mu))R_1(\mu) = N_i(-j\zeta(\mu))R_2(\mu) \tag{2}$$

where N_i is the inner factor of N (see Example 1 below for further details), and ζ, R_1, and R_2 are certain functions of μ. If we search for μ_0 by numerically computing solutions to this equation, computation of the infimal values for H^∞-control criteria will involve the numerical evaluation of the term $N_i(j\omega)$ for different values of ω.

For general transfer functions, one will not be able to explicitly write down an inner-outer factorization, and so the evaluation of the term $N_i(j\omega)$ is not simply a matter of substituting a number into a formula. In this paper we present a numerical technique for this evaluation without explicitly finding the inner factor.

2 Mathematical Framework

The outer factor of an H^∞ function can be computed using the facts that [4, p. 133]

$$|f_o(i\omega)| = |f(i\omega)| \text{ a.e.}$$

$$\arg(f_o(i\omega)) = -\frac{1}{\pi}\int_{-\infty}^{\infty} \frac{\omega t + 1}{(1+t^2)(\omega - t)} \log\left|\frac{f(it)}{f(i\omega)}\right| dt$$

This integral is defined as an ordinary integral, since

$$\int_{-\infty}^{\infty} \frac{\omega t + 1}{(1+t^2)(\omega - t)} \log\left|\frac{f(it)}{f(i\omega)}\right| dt$$

is finite as an ordinary (Lebesgue) integral.

One can thus determine the inner factor numerically, by computing the outer factor using this integral, and then forming the quotient with the original function to obtain the outer factor.

---

[1] This work has been partially supported by the U.S. Air Force Office of Scientific Research under grant No. AFOSR-89-0205.

3 Discussion of Numerical Techniques

Various problems arise in the numerical computation, the most serious of which is the fact that the presence of the logarithm in the kernel may lead to intrinsic computer word-length problems. For large t, the integrand essentially looks like $\frac{\log|f(it)|}{1+t^2}$. Assuming f has a zero at infinity, the numerator becomes unbounded (and negative) for large t, and the integrand is a quotient of two large numbers. Our ability to integrate this numerically depends upon a critical balance between two effects: on the one hand the quotient is getting small, while on the other hand the denominator is getting large. If the rate of increase of the numerator is sufficiently less than that of the denominator, then we can start neglecting the integrand for numerical purposes (setting it to be zero overall) before the size of the denominator exceeds the value allowed by machine wordlength.

In examples, we find that the wordlength is the limiting factor. However it is possible to increase accuracy by an order of magnitude (in numerical experiments) by modeling the error and applying a Richardson-style extrapolation technique.

4 Conclusions

We have presented a numerical technique for the computation of the inner factor of a stable transfer function, intended for use in solving H^∞problems when an explicit factorization is not available. Preliminary computational results show that the technique works, although it is also clear that when the zeros of an infinite Blaschke product are readily computable, this is not the method of choice. Work on applying this method to various models has been done, along with the integration of this computation into a solution for the overall problem addressed by (2). These results will be reported elsewhere. See [3] for a preliminary version.

References

[1] D.S. Flamm, "A Model of a Damped Flexible Beam," ISS Report No. 54,Princeton University Dept. of Electrical Engineering, June 14, 1990.

[2] D.S. Flamm and H. Yang, "H^∞-Optimal Mixed Sensitivity for General Distributed Plants," to appear in *Proc. 29th IEEE Conf. on Decision and Control*, 1990.

[3] D.S. Flamm, H. Yang, Q. Ren and K. Klipec, "Numerical Computation of Inner Factors for Distributed Parameter Systems," ISS Report No. 58, Princeton University Dept. of Electrical Engineering, August 8, 1990.

[4] K. Hoffman, *Banach Spaces of Analytic Functions*, Prentice-Hall, 1962.

[5] K. Klipec, "Derivations of Transfer Functions for Beam Models", September 4, 1990.

[6] G. Zames and S.K. Mitter, "A Note on Essential Spectra and Norms of Mixed Hankel-Toeplitz Operators," *Systems and Control Letters*, 10, 1988, pp. 159-165.

CLOSED-FORM SOLUTIONS TO A CLASS OF H_∞-OPTIMIZATION PROBLEM[1]

Ali Saberi, Ben M. Chen & Zongli Lin
School of Electrical Engineering and Computer Science
Washington State University, Pullman, Washington 99164-2752

ABSTRACT

In this paper, closed-form solutions to a class of H_∞ suboptimal control problem is presented. For the same class of problem the condition under which the optimal H_∞ control problem has a solution is obtained. The explicit expressions for the optimal solutions are also given.

1. INTRODUCTION

Consider a linear time-invariant generalized plant P given in Figure 1 with the control input u, the disturbance w, the measurement output y, and the controlled output z.

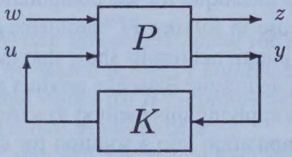


Figure 1: The standard H_∞-optimization problem.

The H_∞ sub-optimal control problem is as follows: for a given number $\gamma > 0$ find a proper control law $u = K(s)y$, if exists, such that the closed-loop system as in given in Figure 1 is internally stable and the H_∞ norm of the transfer function from w to z, denoted by T_{zw}, is less than γ, i.e., $\|T_{zw}\|_\infty < \gamma$. The H_∞-optimal control problem on the other hand is to find a proper control law $u = K_{opt}(s)y$, if exists, such that the closed-loop system is internally stable and $\|T_{zw}\|_\infty$ is minimized. The H_∞ sub-problem control problem has attracted a lot of attention in the last decade. It started with the paper [12]. After that several techniques to solve this problem were developed:

- Interpolation approach: e.g. [6];
- Frequency domain approach: e.g. [3];
- Polynomial approach: e.g. [5];
- J-spectral factorization approach: e.g. [4];
- Time-domain approach: e.g. [2].

The above list is just far from complete. In our view the time domain approach yielded the most intuitive results. All the above techniques have been developed for the so called regular case for which the plant Σ should satisfy the following assumptions:

- The subsystem from control input to the controlled output should not have invariant zeros on the imaginary axis and the direct feedthrough matrix of this system should be injective.
- The subsystem from disturbance to the measurement output should not have invariant zeros on the imaginary axis and the direct feedthrough matrix of this system should be surjective.

---

[1] This work is supported in part by Boeing Commercial Airplane Group and in Part by NASA Langley Research Center under grant contract NAG-1-1210.

The singular case (contrary to the regular case) refers to systems that at least one of the above conditions are not satisfied. Note that identical assumptions were assumed in the linear quadratic Gaussian control problem. The above assumptions for H_∞ suboptimal control problem have recently been removed in [11].

In [2], for the regular case a H_∞ sub-optimal controller has been proposed which is observer-based and can be obtained by solving two parameter-dependent and indefinite algebraic Riccati equations (hereafter referred to as H_∞-ARE's). The solutions for these H_∞-ARE's should also satisfy a certain *coupling condition.* This controller is known as central controller and plays a crucial role in the characterization of all sub-optimal solutions for a given γ. At present time, to obtain any H_∞ sub-optimal controller one must solve these H_∞-ARE's. However, it is well known that these H_∞-ARE's for some values of γ can become highly sensitive and ill-conditioned. This difficulty also arises in the verification of the *coupling condition.* These numerical difficulties are likely to be more severe for the singular case.

The goal of this paper is to provide closed-form solutions to the H_∞ sub-optimal control problem. Here by closed-form solutions we mean solutions which are explicitly parametrized in terms of γ and are obtained without explicitly requiring a value for γ. Hence one can easily tune the parameter γ in order to obtain the desired level of disturbance attenuation. Such a design can be called an 'one-shot' design. We provide these closed-form solutions for a class of singular H_∞ sub-optimal control problem for which the subsystem from the control input u to the controlled output z and the subsystem from the disturbance w to the measurement output y satisfy certain geometric conditions. Moreover, for this class of systems we also provide conditions under which the H_∞ optimal control problem has a solution and explicit expression for the solutions will also be given. Some of the significance of having closed-form solutions in the H_∞ sub-optimal control problem are as follows:

a. In generating closed-form solutions we are not solving any H_∞-ARE's. Therefore, all numerical difficulties associated with H_∞-ARE's have been alleviated.
b. When the closed-form solutions to the H_∞ sub-optimal control problem is implemented either by software or hardware, the value for γ can be adjusted on line. Since the effect of such a 'knob' on the performance and the robustness of the closed-loop system is straightforward, it should be very appealing from a practical point of view.
c. Having closed-form solutions to H_∞ sub-optimal control problem enables us to understand the behavior of the controller (i.e., high-gain, bandwidth, etc.) as the parameter γ approaches to the infimal value of the H_∞ norm of T_{zw} over all stabilizing controllers.

The outline of this paper is as follows. In Section 2 we introduce the problem statement. In Section 3 we provide some preliminaries on the special coordinate basis (s.c.b) and its properties. The s.c.b transformation is instrumental in the derivations of the results given in this paper. Section 4 gives a closed-form solution to the H_∞ sub-optimal state feedback control problem, while Section 5 provides a closed-form solution to H_∞ sub-optimal output feedback control problem. Section 6 draws the conclusions of this paper. Due to space limitation, we omiited Sections 5 and 6 in this manuscript. The complete version of this paper will appear elsewhere.

Throughout this paper we shall adopt the following conventions and notations:

$$
\begin{aligned}
\mathcal{C} &:= \text{whole complex plane.}\\
\mathcal{C}^- &:= \text{open left-half complex plane.}\\
\mathcal{C}^+ &:= \text{open right-half complex plane.}\\
\mathcal{C}^o &:= \text{imaginary axis } j\omega.\\
\lambda(A) &:= \text{the set of eigenvalues of } A.\\
\lambda_{max}(A) &:= \text{maximum eigenvalue of } A \text{ where } \lambda(A) \subset \Re.\\
\operatorname{Ker}(V) &:= \text{kernal of } V.\\
\operatorname{Im}(V) &:= \text{image of } V.
\end{aligned}
$$

We also us define the following subspaces:

i. $\mathcal{V}^g(A, B, C, D)$ — the maximal subspace of $\Re^n$ which is $(A + BF)$–invariant and contained in $\operatorname{Ker}(C + DF)$ such that the eigenvalues of $(A+BF)|\mathcal{V}^g$ are contained in $\mathcal{C}_g \subseteq \mathcal{C}$ for some F.

ii. $\mathcal{S}^g(A, B, C, D)$ — the minimal $(A + KC)$–invariant subspace of $\Re^n$ containing $\operatorname{Im}(B + KD)$ such that the eigenvalues of the map which is induced by $(A + KC)$ on the factor space $\Re^n/\mathcal{S}^g$ are contained in $\mathcal{C}_g \subseteq \mathcal{C}$ for some K.

For the cases that $\mathcal{C}_g = \mathcal{C}$, $\mathcal{C}_g = \mathcal{C}^-$ and $\mathcal{C}_g = \mathcal{C}^o \cup \mathcal{C}^+$, we replace the index g in $\mathcal{V}^g$ and $\mathcal{S}^g$ by '$*$', '$-$' and '$+$' respectively.

2. PROBLEM FORMULATION

Consider a generalized system Σ with a state-space description

$$\Sigma : \begin{cases} \dot{x} = Ax + Bu + Ew \\ y = C_1 x + D_1 w \\ z = C_2 x + D_2 u \end{cases} \tag{2.1}$$

where $x \in \Re^n$ is the state, $u \in \Re^m$ is the control input, $w \in \Re^k$ is the external signal or disturbance, $z \in \Re^\ell$ is the controlled output and $y \in \Re^p$ is the measurement output. The transfer function $P(s)$ from $[w^{\mathrm{T}},\ u^{\mathrm{T}}]^{\mathrm{T}}$ to $[z^{\mathrm{T}},\ y^{\mathrm{T}}]^{\mathrm{T}}$ is

$$P(s) := \begin{bmatrix} P_{11}(s) & P_{12}(s) \\ P_{21}(s) & P_{22}(s) \end{bmatrix} = \left[\begin{array}{c|cc} A & E & B \\ \hline C_2 & 0 & D_2 \\ C_1 & D_1 & 0 \end{array}\right].$$

Following assumptions are made:

Assumption A1 : (A, B) is stablizable;
Assumption A2 : The system (A, B, C_2, D_2) has no invariant zeros in $\mathcal{C}^o$;
Assumption A3 : $\operatorname{Im}(E) \subseteq \mathcal{V}^-(A, B, C_2, D_2) \cup \mathcal{S}^-(A, B, C_2, D_2)$;
Assumption A4 : (A, C_1) is detectable;
Assumption A5 : The system (A, E, C_1, D_1) has no invariant zeros in $\mathcal{C}^o$;
Assumption A6 : $\operatorname{Ker}(C_2) \supseteq \mathcal{V}^-(A, E, C_1, D_1) \cap \mathcal{S}^-(A, E, C_1, D_1)$.

Remark 2.1. *Assumptions A1 and A4, of course, are necessary for the existence of any stabilizing controller. We also would like to mention that if (A, B, C_2, D_2) is right invertible, then A3 holds and if (A, E, C_1, D_1) is left invertible, A6 holds.*

We connect a feedback controller K from y to u as illustrated in Figure 1. The closed-loop transfer function from the disturbance w to the controlled output z is given by the linear fraction map

$$T_{zw} = \mathcal{F}(P, K) = P_{11} + P_{12}K(I - P_{22}K)^{-1}P_{21}.$$

The following definitions will be convenient in the sequel.

Definition 2.1 ((P, γ)-admissible controller). *Let P be a given plant. A proper controller K is said to be (P, γ)-admissible if K stabilizes P and $\|\mathcal{F}(P, K)\|_\infty < \gamma$.*

Definition 2.2 (The infimum of the H_∞ optimal control problem). *The infimum of the H_∞ norm of the closed-loop transfer function $\mathcal{F}(P, K)$ over all stabilizing controllers K denoted by γ_o^*, namely*

$$\gamma_o^* := \inf \{\, \gamma \mid K \text{ is a } (P, \gamma)\text{-admissible controller} \,\}.$$

In the case that the measurement output y is equal to the state (i.e., $C_1 = I$, $D_1 = 0$), we refer to γ_o^* as γ_s^*. It is well known that in this case γ_s^* is equal to the infimum of the H_∞ norm of $\mathcal{F}(P, F)$ over all stabilizing static feedback controllers $u = Fx$.

Definition 2.3 (The H_∞-optimal control problem). *Given a generalized plant P find a proper controller $K_{opt}(s)$ such that $\gamma_o^* = \|\mathcal{F}(P, K_{opt})\|_\infty$.*

The goal of this paper is to obtain a closed-form expressions for a family of (P, γ)-admissible controllers for the class of generalized plants that satisfy Assumptoins A1 – A4. We consider both state feedback and output feedback. Furthermore, for the same class of systems we obtain conditions under which the H_∞ optimal problem has a solution. The explicit expression for the optimal solutions will also be given.

3. A SPECIAL COORDINATE BASIS

In the following we recapitulate the main results in a theorem and some properties of the special coordinate basis while leaving detailed derivation and proofs to be found in [8], [9]. Such a s.c.b has a distinct feature of explicitly displaying the finite and infinite zero structure of a given system. Connections between the s.c.b and the various geometric subspaces of the given system as needed for our development are also given.

Consider the system described by

$$\begin{cases} \dot{x} = Ax + Bu, \\ z = Cx + Du. \end{cases} \tag{3.1}$$

It can be easily shown that using singular value decomposition one can always find an orthogonal transformation U and a nonsingular matrix V that put the direct feedthrough matrix D into the following form

$$\overline{D} = UDV = \begin{bmatrix} I_{m_0} & 0 \\ 0 & 0 \end{bmatrix}, \tag{3.2}$$

where m_0 is the rank of D. Without loss of generality one can assume that the matrix D in equation (3.1) has the form as shown in equation (3.2). Thus the system in (3.1) can be rewritten as

$$\begin{cases} \dot{x} \;=\; A\; x \;+\; [B_0 \quad B_1] \begin{bmatrix} u_0 \\ u_1 \end{bmatrix}, \\ \begin{bmatrix} z_0 \\ z_1 \end{bmatrix} = \begin{bmatrix} C_0 \\ C_1 \end{bmatrix} x + \begin{bmatrix} I_{m_0} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \end{bmatrix}, \end{cases} \tag{3.3}$$

where B_0, B_1, C_0 and C_1 are the matrices of appropriate dimensions. Note that the inputs u_0 and u_1, and the outputs z_0 and z_1 are those of the transformed system. Namely,

$$u = V \begin{bmatrix} u_0 \\ u_1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} z_0 \\ z_1 \end{bmatrix} = Uz.$$

Note that the H_∞-norm of the system transfer function $T_{zw}(s)$ is unchanged when we apply an orthogonal transformation on the output z, and under any nonsingular transformations on the states and control inputs. We have the following main theorem.

Theorem 3.1 (s.c.b). *Consider the given system (A, B, C, D). There exist nonsingular transformations Γ_s, Γ_o and Γ_i, an integer $m_f \le m - m_0$, and integer indices q_i, $i = 1$ to m_f, such that*

$$x = \Gamma_s\, \overline{x}, \quad u = \Gamma_i\, \overline{u}, \quad z = \Gamma_o\, \overline{z}$$

$$\overline{x} = [(x_a^+)^{\mathrm{T}},\; x_b^{\mathrm{T}},\; (x_a^-)^{\mathrm{T}},\; x_c^{\mathrm{T}},\; x_f^{\mathrm{T}}]^{\mathrm{T}}$$

$$x_f = [x_1^{\mathrm{T}},\; x_2^{\mathrm{T}},\; \cdots,\; x_{m_f}^{\mathrm{T}}]^{\mathrm{T}}$$

$$\overline{z} = [z_0^{\mathrm{T}},\; z_f^{\mathrm{T}},\; z_b^{\mathrm{T}}]^{\mathrm{T}}, \quad z_f = [z_1,\; z_2,\; \cdots,\; z_{m_f}]^{\mathrm{T}}$$

$$\overline{u} = [u_0^{\mathrm{T}},\; u_f^{\mathrm{T}},\; u_c^{\mathrm{T}}]^{\mathrm{T}}, \quad u_f = [u_1,\; u_2,\; \cdots,\; u_{m_f}]^{\mathrm{T}},$$

we have the following system of equations:

$$\dot{x}_a^+ = A_{aa}^+ x_a^+ + B_{0a}^+ z_0 + L_{af}^+ z_f + L_{ab}^+ z_b \tag{3.4}$$

$$\dot{x}_b = A_{bb} x_b + B_{0b} z_0 + L_{bf} z_f, \quad z_b = C_b x_b \tag{3.5}$$

$$\dot{x}_a^- = A_{aa}^- x_a^- + B_{0a}^- z_0 + L_{af}^- z_f + L_{ab}^- z_b \tag{3.6}$$

$$\dot{x}_c = A_{cc} x_c + B_{0c} z_0 + L_{cb} z_b + L_{cf} z_f + B_c [E_{ca}^- x_a^- + E_{ca}^+ x_a^+] + B_c u_c \tag{3.7}$$

$$z_0 = C_{0a}^- x_a^- + C_{0a}^+ x_a^+ + C_{0b} x_b + C_{0c} x_c + C_{0f} x_f + u_0 \tag{3.8}$$

and for each $i = 1$ *to* m_f,

$$\dot{x}_i = A_{q_i} x_i + L_{i0} z_0 + L_{if} z_f + B_{q_i} \left[u_i + E_{ia} x_a + E_{ib} x_b + E_{ic} x_c + \sum_{j=1}^{m_f} E_{ij} x_j \right] \tag{3.9}$$

$$z_i = C_{q_i} x_i, \quad z_f = C_f x_f. \tag{3.10}$$

Here the states x_a^-, x_a^+, x_b, x_c *and* x_f *are respectively of dimension* n_a^-, n_a^+, n_b, n_c *and* $n_f = \sum_{i=1}^{m_f} q_i$ *while* x_i *is of dimension* q_i *for each* $i = 1$ *to* m_f. *The control vectors* u_0, u_f *and* u_c *are respectively of dimension* m_0, m_f *and* $m_c = m - m_0 - m_f$ *while the output vectors* z_0, z_f *and* z_b *are respectively of dimension* $p_0 = m_0$, $p_f = m_f$ *and* $p_b = p - p_0 - p_f$. *The matrices* A_{q_i}, B_{q_i} *and* C_{q_i} *have the following form:*

$$A_{q_i} = \begin{bmatrix} 0 & I_{q_i - 1} \\ 0 & 0 \end{bmatrix}, \quad B_{q_i} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad C_{q_i} = [1,\ 0,\ \cdots,\ 0]. \tag{3.11}$$

(Obviously for the case when $q_i = 1$, $A_{q_i} = 0$, $B_{q_i} = 1$ *and* $C_{q_i} = 1$.*) Furthermore, we have* $\lambda(A_{aa}^-) \in \mathcal{C}^-$, $\lambda(A_{aa}^+) \in \mathcal{C}^+$, *the pair* (A_{cc}, B_c) *is controllable and the pair* (A_{bb}, C_b) *is observable. Also, assuming that* x_i *are arranged such that* $q_i \le q_{i+1}$, *the matrix* L_{if} *has the particular form,*

$$L_{if} = [L_{i1},\ L_{i2},\ \cdots,\ L_{i\,i-1},\ 0,\ 0,\ \cdots,\ 0].$$

Also, the last row of each L_{if} *is identically zero.*

Proof : The proof of this theorem can be found in [8] and [9]. ∎

We shall note that the output transformation Γ_o is of form

$$\Gamma_o = \begin{bmatrix} I_{m_0} & 0 \\ 0 & \Gamma_{or} \end{bmatrix}. \tag{3.12}$$

Moreover, we can rewrite the s.c.b given by theorem 3.1 in a more compact form,

$$\begin{cases} \dot{\overline{x}} = \overline{A}\,\overline{x} + \overline{B}\,\overline{u}, \\ \overline{z} = \overline{C}\,\overline{x} + \overline{D}\,\overline{u}, \end{cases} \tag{3.13}$$

where

$$\overline{A} := \Gamma_s^{-1}(A - B_0 C_0)\Gamma_s = \begin{bmatrix} A_{aa}^+ & L_{ab}^+ C_b & 0 & 0 & L_{af}^+ C_f \\ 0 & A_{bb} & 0 & 0 & L_{bf} C_f \\ 0 & L_{ab}^- C_b & A_{aa}^- & 0 & L_{af}^- C_f \\ B_c E_{ca}^+ & B_c E_{cb} & B_c E_{ca}^- & A_{cc} & L_{cf} C_f \\ B_f E_{fa}^+ & B_f E_{fb} & B_f E_{fa}^- & B_f E_{fc} & A_{ff} \end{bmatrix}, \tag{3.14}$$

$$\overline{B} := \Gamma_s^{-1} [\, B_0 \quad B_1 \,] \Gamma_i = \begin{bmatrix} B_{0a}^+ & 0 & 0 \\ B_{0b} & 0 & 0 \\ B_{0a}^- & 0 & 0 \\ B_{0c} & 0 & B_c \\ B_{0f} & B_f & 0 \end{bmatrix}, \quad \overline{D} := \Gamma_o^{-1} D \Gamma_i = \begin{bmatrix} I_{m_0} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.15}$$

and

$$\overline{C} =: \Gamma_o^{-1} \begin{bmatrix} C_0 \\ C_1 \end{bmatrix} \Gamma_s = \begin{bmatrix} C_{0a}^+ & C_{0b} & C_{0a}^- & C_{0c} & C_{0f} \\ 0 & 0 & 0 & 0 & C_f \\ 0 & C_b & 0 & 0 & 0 \end{bmatrix}. \tag{3.16}$$

In what follows, we state some important properties of the s.c.b which are pertinent to our present work.

Property 3.1. *The given system (A, B, C, D) is right-invertible if and only if x_b and hence z_b are nonexistent, left-invertible if and only if x_c and hence u_c are nonexistent, invertible if and only if both x_c and x_b are nonexistent.*

Property 3.2. *Invariant zeros of (A, B, C, D) are the eigenvalues of A_{aa}^- and A_{aa}^+. Moreover, the stable and unstable invariant zeros of (A, B, C, D) are the eigenvalues of A_{aa}^- and A_{aa}^+ respectively.*

Property 3.3. *The pair (A, B) is stabilizable if and only if (A_{con}, B_{con}) is stabilizable where*

$$A_{con} = \begin{bmatrix} A_{aa}^+ & L_{ab}^+ C_b \\ 0 & A_{bb} \end{bmatrix}, \quad B_{con} = \begin{bmatrix} B_{0a}^+ & L_{af}^+ \\ B_{0b} & L_{bf} \end{bmatrix}. \tag{3.17}$$

There are interconnections between the s.c.b and various invariant and almost invariant geometric subspaces. The following property establishs such interconnections.

Property 3.4.

1. $x_a^- \oplus x_a^+ \oplus x_c$ *spans* $\mathcal{V}^*(A, B, C, D)$.
2. $x_a^- \oplus x_c$ *spans* $\mathcal{V}^-(A, B, C, D)$.
3. $x_a^+ \oplus x_c$ *spans* $\mathcal{V}^+(A, B, C, D)$.
4. $x_c \oplus x_f$ *spans* $\mathcal{S}^*(A, B, C, D)$.
5. $x_a^- \oplus x_c \oplus x_f$ *spans* $\mathcal{S}^+(A, B, C, D)$.
6. $x_a^+ \oplus x_c \oplus x_f$ *spans* $\mathcal{S}^-(A, B, C, D)$.

4. STATE FEEDBACK DESIGN

In this section, we consider the case that $y = x$, i.e. consider a system,

$$\Sigma_{\mathrm{P}} : \begin{cases} \dot{x} = Ax + Bu + Ew \\ y = x \\ z = C_2 x + D_2 u. \end{cases} \tag{4.1}$$

It is easy to verify that for this system, Assumptions A4 to A5 are automatically satisfied. Hence, we assume only Assumptions A1 to A3 throughout this section. We introduce a procedure for obtaining the closed-form solutions for the H_∞ sub-optimal state feedback control problem utilizing the asympototic time-scale and eigenstructure assignment (AREA). The concepts of ATEA design procedure was proposed originally in [10]. It uses the spêcial coordinate basis (s.c.b) of the given system (See theorem 3.1). We also give conditions under which the H_∞ optimal control problem has a solution. Furthermore, explicit expressionis for these optimal solutions will be given. In what follows, we gives a step-by-step construction algorithm for the closed-form solution of the suboptimal state feedback laws.

Step 1 : Transform the system (A, B, C_2, D_2) into s.c.b as given by theorem 3.1 in Section 3. To all submatrices and transformations in the s.c.b of (A, B, C_2, D_2), we append the subscript $_{\mathrm{P}}$ to signify their relation to the system (A, B, C_2, D_2). Next we compute

$$\overline{E} = \Gamma_{s\mathrm{P}}^{-1} E = \left[(E_{a\mathrm{P}}^+)^{\mathrm{T}},\ E_{b\mathrm{P}}^{\mathrm{T}},\ (E_{a\mathrm{P}}^+)^{\mathrm{T}},\ E_{c\mathrm{P}}^{\mathrm{T}},\ E_{f\mathrm{P}}^{\mathrm{T}} \right]^{\mathrm{T}}. \tag{4.2}$$

It is simple to verify from the properties of s.c.b. that Assumption A3 implies $E_{b\mathrm{P}} \equiv 0$. Also, for the economy of notation, we denote n_{P} the dimension of $\Re^n/\mathcal{S}^+(A,B,C_2,D_2)$. We not that $n_{\mathrm{P}} = 0$ if and only if the system (A,B,C_2,D_2) is right invertible and is of minimum phase.

If the system (A,B,C_2,D_2) is of nonminimum phase and / or is not right invertible, we define

$$A_{11\mathrm{P}} = \begin{bmatrix} A^+_{aa\mathrm{P}} & L^+_{ab\mathrm{P}}C_{b\mathrm{P}} \\ 0 & A_{bb\mathrm{P}} \end{bmatrix}, \quad B_{11\mathrm{P}} = \begin{bmatrix} B^+_{0a\mathrm{P}} \\ B_{0b\mathrm{P}} \end{bmatrix}, \quad A_{13\mathrm{P}} = \begin{bmatrix} L^+_{af\mathrm{P}} \\ L_{bf\mathrm{P}} \end{bmatrix}$$

$$C_{21\mathrm{P}} = \Gamma_{or\mathrm{P}} \begin{bmatrix} 0 & 0 \\ 0 & C_{b\mathrm{P}} \end{bmatrix}, \quad C_{23\mathrm{P}} = \Gamma_{or\mathrm{P}} \begin{bmatrix} C_{f\mathrm{P}}C'_{f\mathrm{P}} \\ 0 \end{bmatrix}$$

and

$$\begin{aligned} A_{\mathrm{P}} &= A_{11\mathrm{P}} - A_{13\mathrm{P}}(C^{\mathrm{T}}_{23\mathrm{P}}C_{23\mathrm{P}})^{-1}C^{\mathrm{T}}_{23\mathrm{P}}C_{21\mathrm{P}}, \\ B_{\mathrm{P}}B^{\mathrm{T}}_{\mathrm{P}} &= B_{11\mathrm{P}}B^{\mathrm{T}}_{11\mathrm{P}} + A_{13\mathrm{P}}(C^{\mathrm{T}}_{23\mathrm{P}}C_{23\mathrm{P}})^{-1}A^{\mathrm{T}}_{13\mathrm{P}}, \\ C^{\mathrm{T}}_{\mathrm{P}}C_{\mathrm{P}} &= C^{\mathrm{T}}_{21\mathrm{P}}C_{21\mathrm{P}} + C^{\mathrm{T}}_{21\mathrm{P}}C_{21\mathrm{P}}(C^{\mathrm{T}}_{23\mathrm{P}}C_{23\mathrm{P}})^{-1}C^{\mathrm{T}}_{23\mathrm{P}}C_{21\mathrm{P}}. \end{aligned}$$

Then we solve for the unique positive definite solution S_{P} of the algebraic matrix Riccati equation,

$$A_{\mathrm{P}}S_{\mathrm{P}} + S_{\mathrm{P}}A^{\mathrm{T}}_{\mathrm{P}} - B_{\mathrm{P}}B^{\mathrm{T}}_{\mathrm{P}} + S_{\mathrm{P}}C^{\mathrm{T}}_{\mathrm{P}}C_{\mathrm{P}}S_{\mathrm{P}} = 0 \tag{4.3}$$

together with the matrix T_{P} defined by

$$T_{\mathrm{P}} = \begin{bmatrix} T_{aa\mathrm{P}} & 0 \\ 0 & 0 \end{bmatrix} \tag{4.4}$$

where $T_{aa\mathrm{P}}$ is the unique semi-positive solution of the algebraic matrix Lyapunov equation,

$$A^+_{aa\mathrm{P}}T_{aa\mathrm{P}} + T_{aa\mathrm{P}}(A^+_{aa\mathrm{P}})^{\mathrm{T}} = E^+_{a\mathrm{P}}(E^+_{a\mathrm{P}})^{\mathrm{T}}. \tag{4.5}$$

Note that A^+_{aa} is anti-stable and $(A_{bb\mathrm{P}}, C_{b\mathrm{P}})$ is observable imply the pair $(-A_{\mathrm{P}}, C_{\mathrm{P}})$ is detectable, and Assumption A1 implies the pair $(A_{\mathrm{P}}, B_{\mathrm{P}})$ is stabilizable. Hence the existence and uniqueness of the solutions for S_{P} and $T_{aa\mathrm{P}}$ follow from the results of [7]. Then it is shown in Chen, Saberi and Ly [1] that the infimum of the H_∞ optimal control problem for Σ_{P}, i.e. γ^*_s, is given by

$$\gamma^*_s = \begin{cases} \sqrt{\lambda_{max}(T_{\mathrm{P}}S^{-1}_{\mathrm{P}})} & \text{if } n_{\mathrm{P}} > 0, \\ 0 & \text{if } n_{\mathrm{P}} = 0. \end{cases} \tag{4.6}$$

Step 2 : Let F_c be any arbitrary $m_{c\mathrm{P}} \times n_{c\mathrm{P}}$ matrix subject to the constraint that $A^c_{cc\mathrm{P}} = A_{cc\mathrm{P}} - B_{c\mathrm{P}}F_c$, is a stable matrix. Note that the existence of such an F_c is guaranteed by the property of s.c.b, i.e. $(A_{cc\mathrm{P}}, B_{c\mathrm{P}})$ is controllable.

Step 3 : Next, given any $\gamma \geq \bar{\gamma} > \gamma^*_s$, we define

$$F_{11}(\bar{\gamma}) := \begin{bmatrix} F^+_{a0}(\bar{\gamma}) & F_{b0}(\bar{\gamma}) \\ F^+_{a1}(\bar{\gamma}) & F_{b1}(\bar{\gamma}) \end{bmatrix} = \begin{bmatrix} B^{\mathrm{T}}_{11\mathrm{P}}P_0(\bar{\gamma}) \\ (C^{\mathrm{T}}_{23\mathrm{P}}C_{23\mathrm{P}})^{-1}[A^{\mathrm{T}}_{13\mathrm{P}}P_0(\bar{\gamma}) + C^{\mathrm{T}}_{23\mathrm{P}}C_{21\mathrm{P}}] \end{bmatrix} \tag{4.7}$$

where

$$P_0(\bar{\gamma}) := (S_{\mathrm{P}} - \bar{\gamma}^{-2}T_{\mathrm{P}})^{-1} \tag{4.8}$$

and define

$$A^c_{11\mathrm{P}} := A_{11\mathrm{P}} - [B_{11\mathrm{P}}, A_{13\mathrm{P}}]F_{11}(\bar{\gamma}).$$

As we will show later on that the eigenvalues of $A^c_{11\mathrm{P}}$ are in $\mathcal{C}^-$. Moreover, let us partition $[\,F^+_{a1}(\bar{\gamma}) \;\; F_{b1}(\bar{\gamma})\,]$ as,

$$[\,F^+_{a1}(\bar{\gamma}) \;\; F_{b1}(\bar{\gamma})\,] = \begin{bmatrix} F^+_{a11}(\bar{\gamma}) & F_{b11}(\bar{\gamma}) \\ F^+_{a12}(\bar{\gamma}) & F_{b12}(\bar{\gamma}) \\ \vdots & \vdots \\ F^+_{a1m_{f\mathrm{P}}}(\bar{\gamma}) & F_{b1m_{f\mathrm{P}}}(\bar{\gamma}) \end{bmatrix} \tag{4.9}$$

where $F_{a1i}^{+}(\bar{\gamma})$ and $F_{b1i}(\bar{\gamma})$ are of dimension $1 \times n_{a\mathrm{P}}^{+}$ and $1 \times n_{b\mathrm{P}}$, respectively.

Step 4 : This step makes use of subsystems, $i = 1$ to $m_{f\mathrm{P}}$, represented by (3.9) of Section 3. Let $\Lambda_i = \{\, \lambda_{i1},\ \lambda_{i2},\ \cdots,\ \lambda_{iq_i}\, \}$, $i = 1$ to $m_{f\mathrm{P}}$, be the set of q_i elements all in $\mathcal{C}^-$, which are closed under complex conjugation, where q_i and $m_{f\mathrm{P}}$ are as defined in theorem 3.1 but associated with the s.c.b of (A, B, C_2, D_2). Let $\Lambda_{f\mathrm{P}} := \Lambda_1 \cup \Lambda_2 \cup \cdots \cup \Lambda_{m_{f\mathrm{P}}}$. For $i = 1$ to $m_{f\mathrm{P}}$, we define

$$p_i(s) := \prod_{j=1}^{q_i} (s - \lambda_{ij}) = s^{q_i} + F_{i1} s^{q_i - 1} + \cdots + F_{iq_i - 1} s + F_{iq_i} \tag{4.10}$$

and

$$\tilde{F}_i(\epsilon, \Lambda_i) := \frac{1}{\epsilon^{q_i}} \Big[F_{iq_i},\ \epsilon F_{iq_i - 1},\ \cdots,\ \epsilon^{q_i - 1} F_{i1} \Big]. \tag{4.11}$$

Step 5 : In this step, various gains calculated in Steps 2 to 4 are put together to form a composite state feedback gain for the given system Σ. Let

$$\Big[\tilde{F}_{a1}^{+}(\bar{\gamma}, \epsilon, \Lambda_{f\mathrm{P}})\ \ \tilde{F}_{b1}(\bar{\gamma}, \epsilon, \Lambda_{f\mathrm{P}}) \Big] := \begin{bmatrix} F_{a11}^{+}(\bar{\gamma}) F_{1q_1} / \epsilon^{q_1} & F_{b11}(\bar{\gamma}) F_{1q_1} / \epsilon^{q_1} \\ F_{a12}^{+}(\bar{\gamma}) F_{2q_2} / \epsilon^{q_2} & F_{b12}(\bar{\gamma}) F_{2q_2} / \epsilon^{q_2} \\ \vdots & \vdots \\ F_{a1m_{f\mathrm{P}}}^{+}(\bar{\gamma}) F_{m_{f\mathrm{P}} q_{m_{f\mathrm{P}}}} / \epsilon^{q_{m_{f\mathrm{P}}}} & F_{b1m_{f\mathrm{P}}}(\bar{\gamma}) F_{m_{f\mathrm{P}} q_{m_{f\mathrm{P}}}} / \epsilon^{q_{m_{f\mathrm{P}}}} \end{bmatrix}.$$

Then define the state feedback gain $F(\bar{\gamma}, \epsilon, \Lambda_{f\mathrm{P}}, F_c)$ as

$$F(\bar{\gamma}, \epsilon, \Lambda_{f\mathrm{P}}, F_c) = -V_{\mathrm{P}} \Gamma_{i\mathrm{P}} \tilde{F}(\bar{\gamma}, \epsilon, \Lambda_{f\mathrm{P}}, F_c) \Gamma_{s\mathrm{P}}^{-1} \tag{4.12}$$

where

$$\tilde{F}(\bar{\gamma}, \epsilon, \Lambda_{f\mathrm{P}}, F_c) = \\ \begin{bmatrix} C_{0a\mathrm{P}}^{+} + F_{a0}^{+}(\bar{\gamma}) & C_{0b\mathrm{P}} + F_{b0}(\bar{\gamma}) & C_{0a\mathrm{P}}^{-} & C_{0c\mathrm{P}} & C_{0f\mathrm{P}} \\ E_{fa\mathrm{P}}^{+} + \tilde{F}_{a1}^{+}(\bar{\gamma}, \epsilon, \Lambda_{f\mathrm{P}}) & E_{fb\mathrm{P}} + \tilde{F}_{b1}(\bar{\gamma}, \epsilon, \Lambda_{f\mathrm{P}}) & E_{fa\mathrm{P}}^{-} & E_{fc\mathrm{P}} & \tilde{F}_f(\epsilon, \Lambda_{f\mathrm{P}}) + E_{f\mathrm{P}} \\ E_{ca\mathrm{P}}^{+} & E_{cb\mathrm{P}} & E_{ca\mathrm{P}}^{-} & F_c & 0 \end{bmatrix}$$

and where

$$E_{f\mathrm{P}} = \begin{bmatrix} E_{11} & \cdots & E_{1m_{f\mathrm{P}}} \\ \vdots & \ddots & \vdots \\ E_{m_{f\mathrm{P}}1} & \cdots & E_{m_{f\mathrm{P}} m_{f\mathrm{P}}} \end{bmatrix}$$

and

$$\tilde{F}_f(\epsilon, \Lambda_{f\mathrm{P}}) = \mathrm{Diag}\Big[\tilde{F}_1(\epsilon, \Lambda_1),\ \tilde{F}_2(\epsilon, \Lambda_2),\ \cdots,\ \tilde{F}_{m_f}(\epsilon, \Lambda_i) \Big].$$

We have the following theorem.

Theorem 4.1. *Consider a given system Σ_{P} satisfying Assumptions A1 to A3. Then with state feedback gain given by (4.12), we have the following properties:*

1. *For any $\gamma_s^* < \bar{\gamma} \le \gamma$, for any $\Lambda_{f\mathrm{P}} \subset \mathcal{C}^-$ which is closed under complex conjugation and for any $F_c \in \Re^{m_{c\mathrm{P}} \times n_{c\mathrm{P}}}$ subject to $A_{cc\mathrm{P}}^{c}$ is a stable matrix, there exists $\epsilon^* > 0$ such that for all $\epsilon \le \epsilon^*$, the closed-loop is internally stable and the H_∞-norm of the closed-loop transfer function from the disturbance w to the controlled output z is less than γ, i.e.*

$$\|T_{zw}\|_\infty < \bar{\gamma} \le \gamma,$$

where $T_{zw} = \mathcal{F}(P, F(\bar{\gamma}, \epsilon, \Lambda_{f\mathrm{P}}, F_c))$.

2. *Moreover as $\epsilon \to 0$, eigenvalues of the closed-loop system, i.e. the eigenvalues of $A - BF(\bar{\gamma}, \epsilon, \Lambda_{f\mathrm{P}}, F_c)$, are given by*

$$\lambda(A^{-}_{aa\mathrm{P}}), \quad \lambda(A^{c}_{cc\mathrm{P}}), \quad \lambda(A^{c}_{11\mathrm{P}}) + 0(\epsilon) \quad \text{and} \quad \frac{\Lambda_{f\mathrm{P}}}{\epsilon} + 0(1).$$

Proof : Due to space limitation, the proof is omitted. ∎

The following remarks are in order.

Remark 4.1 (Interpretations of parameters ϵ, $\Lambda_{f\mathrm{P}}$ and F_c). *Theorem 4.1 shows that the closed-loop system under H_∞ sub-optimal state feedback law, i.e. T_{zw}, has fast eigenvalues $\Lambda_{f\mathrm{P}}/\epsilon$. So the set parameter $\Lambda_{f\mathrm{P}}$ in the H_∞ sub-optimal gain $F(\bar{\gamma}, \epsilon, \Lambda_{f\mathrm{P}}, F_c)$ of (4.12) represents the asymptoties of these fast eigenvalues while ϵ represents their time-scale. The closed-loop system also has $\lambda(A^{c}_{cc\mathrm{P}})$ as slow eigenvalues. These eigenvalues can be assigned to any desired locations in C^- by choosing an appropriate F_c. Hence, the set parameter F_c in the H_∞ sub-optimal state feedback gain represents the locations of these slow eigenvalues.*

Remark 4.2 (Pole and zero cancellations). *Theorem 4.1 also shows that the closed-loop system T_{zw} has poles at $\lambda(A^{c}_{cc\mathrm{P}})$ and $\lambda(A^{-}_{aa\mathrm{P}})$. Note that $\lambda(A^{-}_{aa\mathrm{P}})$ are the stable invariant zeros of (A, B, C_2, D_2). It is simple to show that $\lambda(A^{c}_{cc\mathrm{P}})$ and $\lambda(A^{-}_{aa\mathrm{P}})$ are the output decoupling zeros of T_{zw} as well. Hence, they cancel with the poles of T_{zw}.*

Remark 4.3 (Regular case). *If D_2 is injective, it is obvious to verify from our algorithm that $F(\bar{\gamma}, \epsilon, \Lambda_{f\mathrm{P}}, F_c) = F(\bar{\gamma})$ does not depend on ϵ, $\Lambda_{f\mathrm{P}}$ and F_c, and is given by*

$$F(\bar{\gamma}) = -V_{\mathrm{P}}\Gamma_{i\mathrm{P}}\left[C^{+}_{0a\mathrm{P}} + F^{+}_{a0}(\bar{\gamma}) \quad C_{0b\mathrm{P}} + F_{b0}(\bar{\gamma}) \quad C^{-}_{0a\mathrm{P}}\right]\Gamma^{-1}_{s\mathrm{P}}.$$

This corresponds to the regular case and for $\bar{\gamma} = \gamma$, is the central controller given in [2]. Moreover, if $\bar{\gamma} = \infty$, the result reduces to the well-known LQG solution.

In next theorem, we show the conditions under which the H_∞ optimal state feedback control problem has a solution. Furthermore, we examine the behavior of the H_∞ sub-optimal solutions as $\gamma \to \gamma_s^*$.

Theorem 4.2.

1. *The H_∞ optimal controller exists if and only if $\operatorname{Im}(E) \subseteq \mathcal{S}^-(A, B, C_2, D_2)$. Moreover, in this case $\gamma_s^* = 0$.*
2. *If $\operatorname{Im}(E) \not\subseteq \mathcal{S}^-(A, B, C_2, D_2)$, any static or dynamic state feedback*

$$u = \left[C_{cmp}(sI - A_{cmp})^{-1}B_{cmp} + D_{cmp}\right]x,$$

which is (P, γ)-admissible will exhibit high-gain, i.e.

$$\left\| \begin{bmatrix} A_{cmp} & B_{cmp} \\ C_{cmp} & D_{cmp} \end{bmatrix} \right\| \to \infty, \quad \text{as } \gamma \to \gamma_s^*.$$

Proof : Due to space limitation, the proof is omitted. ∎

In what follows we will provide a construction algorithm that gives a set of the H_∞ optimal state feedback laws.

Step 1 : Transform the system (A, B, C_2, D_2) in the form of s.c.b.

Step 2 : Let F_c be any arbitrary $m_{c\mathrm{P}} \times n_{c\mathrm{P}}$ matrix subject to the constraint that $A^{c}_{cc\mathrm{P}} = A_{cc\mathrm{P}} - B_{c\mathrm{P}}F_c$ is asymptotically stable.

Step 3 : Form matrices A_x and B_x as follows:

$$A_x := \begin{bmatrix} A^+_{aa\mathrm{P}} & L^+_{ab\mathrm{P}} C_{b\mathrm{P}} & L^+_{af\mathrm{P}} C_{f\mathrm{P}} \\ 0 & A_{bb\mathrm{P}} & L_{bf\mathrm{P}} C_{f\mathrm{P}} \\ B_{f\mathrm{P}} E^+_{fa\mathrm{P}} & B_{f\mathrm{P}} E_{fb\mathrm{P}} & A_{ff\mathrm{P}} \end{bmatrix}, \quad B_x := \begin{bmatrix} B^+_{a0\mathrm{P}} & 0 \\ B_{b0\mathrm{P}} & 0 \\ B_{f0\mathrm{P}} & B_{f\mathrm{P}} \end{bmatrix}.$$

Let F_x be any arbitrary $(m_{0\mathrm{P}} + m_{f\mathrm{P}}) \times (n^+_{a\mathrm{P}} + n_{b\mathrm{P}} + n_{f\mathrm{P}})$ matrix subject to the constraint that $A^c_x = A_x - B_x F_x$ is asymptotically stable. Here we note that it is simple to verify from the properties of s.c.b that the pair (A_x, B_x) is detectable if and only if (A, B) is detectable. Hence, such a F_x always exists. Next partition F_x as,

$$F_x = \begin{bmatrix} F^+_{a0} & F_{b0} & F_{f0} \\ F^+_{a1} & F_{b1} & F_{f1} \end{bmatrix}.$$

Step 4 : Let

$$F = -V_{\mathrm{P}} \Gamma_{i\mathrm{P}} \begin{bmatrix} C^+_{0a\mathrm{P}} + F^+_{a0} & C_{0b\mathrm{P}} + F_{b0} & C^-_{0a\mathrm{P}} & C_{0c\mathrm{P}} & C_{0f\mathrm{P}} + F_{f0} \\ F^+_{a1} & F_{b1} & E^-_{fa\mathrm{P}} & E_{fc\mathrm{P}} & F_{f1} \\ F^+_{ca} & F_{cb} & F^-_{ca} & F_c & F_{cf} \end{bmatrix} \Gamma^{-1}_{s\mathrm{P}}, \tag{4.13}$$

where F^+_{ca}, F_{cb}, F^-_{ca} and F_{cf} are some arbitrary submatrices with approprite dimension.

We have the following result.

Theorem 4.3. *Given a system Σ_{P} which satisfies the condition $\mathrm{Im}\,(E) \subseteq \mathcal{S}^-(A, B, C_2, D_2)$, i.e. the H_∞ optimal problem has a solution. Then for any state feedback law $u = Fx$ with F given by (4.13), the closed-loop is internally stable and the H_∞-norm of the closed-loop transfer function from w to z is equal to $\gamma^*_s = 0$. Moreover, eigenvalues of the closed-loop system are given by $\lambda(A^-_{aa\mathrm{P}})$, $\lambda(A^c_{cc\mathrm{P}})$ and $\lambda(A^c_x)$. In fact, if $\mathrm{Im}\,(E) = \mathcal{S}^-(A, B, C_2, D_2)$, any stabilizing state feedback gain F which achieves the infimum is of the form (4.13).*

Proof : Due to space limitation, the proof is omitted. ∎

References

[1] B. M. Chen, A. Saberi and U. Ly, "A non-iterative method for computing the infimum in H_∞-optimization," To be presented in *the 30th CDC*, 1991.

[2] J. Doyle, K. Glover, P. P. Khargonekar and B. A. Francis, *IEEE Transactions on Automatic Control*, Vol. 34, pp. 831-847, 1988.

[3] B. A. Francis, *A course in H_∞ control theory*, Vol. 88 of LNCIS, Springer-Verlag, 1987.

[4] H. Kimura, *International Journal of Control*, Vol. 49, pp. 269-307, 1989.

[5] H. Kwakernaak, *International Journal of Control*, Vol. 41, pp. 117-156, 1986.

[6] D. J. N. Limbeer and B. D. O. Anderson, *Linear Algebra and its Applications*, Vol. 98, pp. 347-386, 1988.

[7] T. J. Richardson and R. H. Kwong, *Systems & Control Letters*, 7, 99-104, 1986.

[8] P. Sannuti and A. Saberi, *International Journal of Control*, Vol. 45, pp. 1655-1704, 1987.

[9] A. Saberi and P. Sannuti, *International Journal of Control*, Vol. 51, pp. 621-629, 1990.

[10] A. Saberi and P. Sannuti, *International Journal of Control*, Vol. 49, pp. 2191-2213, 1989.

[11] A. A. Stoorvogel, *SIAM Journal on Control & Optimization*, Vol. 29, pp. 160-184, 1991.

[12] G. Zames, *IEEE Transactions on Automatic Control*, Vol 26, pp. 301-320, 1981.

ASSESSING THE EFFECTS OF PLANT DYNAMIC VARIATIONS ON THE ROBUSTNESS AND PERFORMANCE OF FEED FORWARD FEEDBACK

David Bensoussan, Maher Hammami
Département de Génie électrique
École de technologie supérieure
Université du Québec
4750, Henri-Julien
Montréal (Québec) Canada H2T 2C8

ABSTRACT

One of the main reasons to gear control theory towards H_∞ analysis and synthesis, is that such theory handles nicely plant variations while state space type solutions (linear quadratic methods) always assume the exact knowledge of the state space matrices [A,B,C,D]. The state space matrix uncertainty problem is considered here in conjunction with its frequency response counterpart. It is proposed in this paper to study a feedforward feedback structure while insuring that the resulting control remains stable under plant dynamic variations ΔA. Bounds on the norm of ΔA are derived so that all the feedback H_∞ transmissions norms are stable and bounded.

I. INTRODUCTION

Feedforward control advantages have been known [1] and the studies of robustness under that scheme have been studied [2]. However, the later results have assumed that a plant frequency response P(s) is modified to $P(s) + \Delta P(s)$. Although it is convenient to describe $\Delta P(s)$ as a ball of uncertainty, it is nevertheless difficult to quantify. On the other hand, in dynamic systems described by differential equations, some of the coefficients appearing in a differential equation might represent physical entities which are more subject to be modified than others. It has been proposed [3,4] to relate individual variations δa_{ij} of the entries a_{ij} of the state space model of the plant dynamic matrix A to the matrix variation ΔA as follows. Let $\hat{A}$ be the $n \times n$ matrix whose elements are the induced L_2 norms $\| a_{ij} \|$. We define a new norm $\| | A \| |$ to be the maximum singular value of the new matrix $\hat{A}$. It has been shown [3] that such a norm is compatible to the standard induced L_2 norm as follows : $\|A\| \leq \| |A\| | \leq n \|A\|$.

Similarly of a $n \times n$ frequency response P(s), whose elements are $P_{ij}(s)$, we define $\hat{P}$ to be the matrix whose entries are $\| P_{ij}(s) \|_\infty$ and $\| | P \| |$ to be the maximal singular value of $\hat{P}$ so that : $\|P\|_\infty \leq \| |P\| | \leq n \|P\|_\infty$.

In the following, we will study the effect of ΔA on the stability of a transfer function (Section 2). We then extend the result to a feedforward feedback control scheme (Section 3), and to the general problem of state/output feedback with state dependant noise (Section 4).

II. STUDYING THE EFFECT OF PLANT DYNAMIC VARIATIONS

Let the state space model of a plant P(s) be described by the matrices $[A_P, B_P, C_P, D_P]$ so that : $\dot{x} = A_P x + B_p u + \sigma$, $y = C_P x + D_p u$

Consider the perturbed system : $\dot{x} = (A_P + \Delta A) x + B_p u + \sigma$, $y = C_P x + D_p u$

The output functions T(s) and T'(s), and disturbance to output transfer function $T_\sigma(s)$ and $T'_\sigma(s)$ are defined as follows :

$$\begin{aligned} T(s) &= C_P(sI - A_P)^{-1} B_P + D_P \\ T'(s) &= C_P(sI - A_P - \Delta A)^{-1} B_P + D_P \\ T_\sigma(s) &= C_P(sI - A_P)^{-1} \\ T'_\sigma(s) &= C_P(sI - A_P - \Delta A)^{-1} \end{aligned}$$

It has been shown [3] that :

$$\| T'(s) - T(s) \| \leq K_P \frac{\| \Delta A \|}{1 - \| (sI - A_P)^{-1} \| . \| \Delta A \|} \tag{1}$$

where $K_p = \| C_p(sI - A)_p)^{-1} \| \cdot \| (sI - A_p)^{-1} B \|$ and where $\| (sI - A_P)^{-1} \Delta A \| < 1$ is assumed. A similar variation can be derived for $\| T'_\sigma(s) - T_\sigma(s) \|$.

Relation (1) is a monotonous increasing function of $\| \Delta A \|$. It represents an upper bound on $\| T'(s) - T(s) \|$. This bound is asymptotic to the line:

$$\| \Delta A \| = \frac{1}{\| (sI - A_p)^{-1} \|}$$

III. THE GENERAL FEEDBACK CASE

Referring to Figure 2, it is possible to show that, given a plant P, a series compensation C, a feedback compensation F, and the feedforward compensations Q and G described by their state space matrices $[A_P, B_P, C_P, D_P]$, $[A_C, B_C, C_C, D_C]$, $[A_F, B_F, C_F, D_F]$, $[A_G, B_G, C_G, D_G]$, $[A_Q, B_Q, C_Q, D_Q]$, it is possible to relate the state space descriptions of the transmissions of the feedback loop in which d_1 d_2 and d_3 represent additive disturbance to the output, sensor noise and plant output noise and in which r represents the input as follows :

$$\begin{bmatrix} y \\ e \\ v \end{bmatrix} = \begin{bmatrix} (I+PCF)^{-1} & & \\ & (I+FPC)^{-1} & \\ & & (I+CFP)^{-1} \end{bmatrix} \begin{bmatrix} P(CQ+G) & I & -PCF & P \\ (Q-FPG) & -F & -F & -FP \\ CQ+G & -CF & -CF & I \end{bmatrix} \begin{bmatrix} r \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} \tag{2}$$

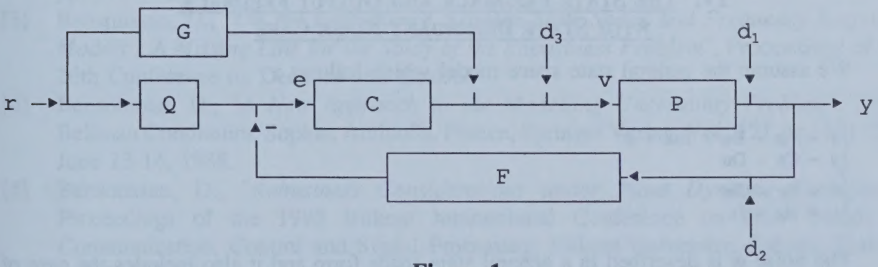


Figure 1

Let us denote by H_{ij} the transmissions described in (2), where i = 1,2,3 and j = 1,2,3,4, so that $H_{11} = (I+PCF)^{-1} P(CQ+G)$, $H_{23} = -(I+FPC)^{-1}F$ and so forth. In the following, we will identify the state space descriptions of all the transmissions of the matrix H so that to simplify the study of plant dynamics variations ΔA on each transmission. It will be assumed in the following that the inverses of the operators described in (2) do exist.

We will start by identifying the state space descriptions of the following transmissions.

$$E = (I + PCF)^{-1} = \left[\begin{array}{ccc|c} A_F - B_F Y_{PCF} D_P D_C C_F & -B_F Y_{PCF} D_P C_C & -B_F Y_{PCF} C_P & B_F Y_{PCF} \\ B_C Y_{FPC} C_F & A_C - B_C Y_{FPC} D_F D_P C_C & -B_C Y_{FPC} D_F C_P & B_C Y_{FPC} D_F \\ B_P Y_{CFP} D_C C_F & B_P Y_{CFP} C_C & A_P - B_P Y_{CFP} D_C D_F C_P & B_P Y_{CFP} D_C D_F \\ \hline -Y_{PCF} D_P D_C C_F & -Y_{PCF} D_P C_C & -Y_{PCF} C_P & Y_{PCF} \end{array}\right]$$

$$E' = (I + FPC)^{-1} = \left[\begin{array}{ccc|c} A_C - B_C Y_{FPC} D_F D_P C_C & -B_C Y_{FPC} D_F C_P & -B_C Y_{FPC} C_F & B_C Y_{FPC} \\ B_P Y_{CFP} C_C & A_P - B_P Y_{CFP} D_C D_F C_P & -B_P Y_{CFP} D_C C_F & B_P Y_{CFP} D_C \\ B_F Y_{PCF} D_P C_C & B_F Y_{PCF} C_P & A_F - B_F Y_{PCF} D_P D_C C_F & B_F Y_{PCF} D_P D_C \\ \hline -Y_{FPC} D_F D_P C_C & -Y_{FPC} D_F C_P & -Y_{FPC} C_F & Y_{FPC} \end{array}\right]$$

$$E'' = (I + CFP)^{-1} = \left[\begin{array}{ccc|c} A_P - B_P Y_{CFP} D_C D_F C_P & -B_P Y_{CFP} D_C C_F & -B_P Y_{CFP} C_C & B_P Y_{CFP} \\ B_F Y_{PCF} C_P & A_F - B_F Y_{PCF} D_P D_C C_F & -B_F Y_{PCF} D_P C_C & B_F Y_{PCF} D_P \\ B_C Y_{FPC} D_F C_P & B_C Y_{FPC} C_F & A_C - B_C Y_{FPC} D_F D_P C_C & B_C Y_{FPC} D_F D_P \\ \hline -Y_{CFP} D_C D_F C_P & -Y_{CFP} D_C C_F & -Y_{CFP} C_C & Y_{CFP} \end{array}\right]$$

with $Y_{FPC} \triangleq (I + D_F D_P D_C)^{-1}$, $Y_{PCF} \triangleq (I + D_P D_C D_F)^{-1}$, $Y_{CFP} \triangleq (I + D_C D_F D_P)^{-1}$

It follows that H_{ij} can be deduced from the state matrices of the following product according to the rules pertaining for state space representation of sum, product and inverses of state space models.

We note that in all transmissions of H_{ij} it is possible to isolate A_p so as to consider in the new perturbed matrix H'_{ij} nominal plant A_p and the disturbed plant $A_p + \Delta A$. Relation (1) can therefore, be extrapolated so that we derive bounds on $\| H'_{ij}(s) - H_{ij}(s) \|$ where H_{ij} is a nominal transmission and $H_{ij}(s)$ is the transmission under plant dynamic variations.

IV. THE STATE FEEDBACK AND OUTPUT FEEDBACK WITH STATE DEPENDANT NOISE CASE

We assume the general state space model which follows :

$$\begin{cases} \dot{x} = Ax + Bu + Ew + \sigma \\ y = Cx + Du \end{cases}$$

$$\begin{cases} \dot{z} = Fz + Gx \\ w = Hz + Jx \end{cases}$$

The noise w is described in a general state space form and it also includes the case of state dependant noise. Such a noise representation is convenient as a state dependant noise can model parameter variations [6]. The Ew factor may represent a stochastic parameter variation of the system matrix A while σ represents an external additive perturbation.

We define the new state space variable $\tilde{x} = \begin{pmatrix} x \\ z \end{pmatrix}$. It then follows the system can be described with the following augmented matrices.

$$\dot{\tilde{x}} = \begin{pmatrix} A+EJ & EH \\ G & F \end{pmatrix} \tilde{x} + \begin{pmatrix} B \\ O \end{pmatrix} u \tag{3}$$

$$y = (C \quad O) \;\; \tilde{x} + Du$$

which we represent by (4), (5) in the case of state feedback applied to augmented matrix, and (6) in the case of output feedback applied to the augmented matrix.

$$\left[\begin{array}{cc|c} A+EJ & EH & B \\ \underline{G} & \underline{F} & \underline{O} \\ C & O & D \end{array}\right] \quad (4) \qquad \left[\begin{array}{cc|c} A+EJ-BK_s & EH & B \\ \underline{\quad G \quad} & \underline{F} & \underline{O} \\ C & O & D \end{array}\right] \quad (5)$$

$$\left[\begin{array}{ccc|c} A+EJ-BK_cC & EH & & \begin{pmatrix} B \\ O \end{pmatrix}(I-K_oD) \\ \underline{\quad G \quad} & \underline{F} & \underline{\quad} & \underline{\qquad} \\ C & O & & D \end{array}\right] \quad (6)$$

In (3), (4), (5) and (6), A can be replaced by A + ΔA and the relations developed in Section 1 can be applied. ΔA might also include the matrix EJ, EH, G and F as a random perturbation of A but not exclusively.

REFERENCES

[1] Horowitz, I., "*Synthesis of Feedback Systems*", New York, Academic Press, 1963.

[2] Hara, S., "*Parametization of Stabilizing Controllers for Multivariable Servo Systems with Two Degrees of Freedom*", International Journal of Control, 1987, Vol. 45, No. 3, pp. 779-790.

[3] Bensoussan, D., "*On the Equivalence Between State Space and Frequency Response Models : A Missing Link for the Study of the Robustness Problem*", Proceedings of the 26th Conference on Decision and Control.

[4] Bensoussan, D., "*A New Approach to the Modelling Uncertainty Problem*", Third Bellman Continuum, Sophia, Antipolis, France, Springer Verlag, Vol. 121, pp. 121-129, June 13-14, 1988.

[5] Bensoussan, D., "*Robustness Considerations under Plant Dynamic Variations*", Proceedings of the 1990 Bilkent International Conference on New Trends in Communication, Control and Signal Processing, Bilkent University, Ankara, Turkey, July 2-5, 1990.

[6] Wonham, W.M., "*Optimal Stationary Control of a Linear System with State Dependant Noise*", SIAM Journal of Control, Vol. 5, August 1967, pp. 486-500.

A Time Varying Nehari Theorem: A System Theoretic Approach

Gilead Tadmor[1] and Madanpal Verma[2]

Abstract. Nehari Theorem is interpreted as providing a measure of the distance of a bounded I/O operator from the family of causal bounded I/O operators. A time varying variant is established, along with an analysis of invariant stable — anti-stable state-space partitions and co-prime factorzations in LTV systems.

0 Introduction

From both the conceptual and the computational perspectives, Nehari Theorem (Nehari, 1957) and its later extensions (notably, e.g. Adamjan, Arov & Krein 1971, Ball & Helton 1983, Dym 1989, Power 1982, Sarason 1967, Sz.-Nagy & Foias 1969) are fundamental in modern system theory. In its original form, it evaluates the minimum distance of a meromorphic function from the class of analytic functions over a complex domain. Referring to transfer functions of LTI systems, this setting is directly related to model matching problems, and other design issues that can be translated into a model matching form [F].

In this note we suggest a time-domain state-space interpretation of Nehari Theorem, which extends to the time-varying case. We establish the extended theorem with heavy reliance on the technical tool-box of the recent time-domain variational approach to H_∞ control (see e.g Tadmor 1988, 1989a,b, 1990a,b) as well as on previous LTI and LTV discussions (Ball & Ran 1987, Ball & al. 1988, Feintuch & Francis 1985, Glover 1986, Gohberg & al 1989a,b, Verma 1989). The discussion brings to bear issues of independent interest, regarding state-space analysis of LTV systems. Particularly, we analyze invariant stable — anti-stable state space partitions and aspects of co-prime factorizations.

The dichotomy between "traditional" factorization based frequency-domain H_∞ techniques and the more recent variational time-domain tools, is intriguing. It is of natural interest to attempt to reconcile the two approaches. This work represents a step in that direction. It is also motivated by the computational advantage of using Nehari Theorem in the solution of a one block model-matching problem, saving the need for an iterative search for the optimal value, as in the general case of the standard H_∞ problem.

1 Some Basic Concepts

Consider a finite-dimensional time varying, bounded coefficients, linear system

$$\begin{aligned} \dot{x}(t) &= A(t)x(t) + B(t)u(t), \quad t \in \Re, x \in \Re^n \\ y(t) &= C(t)x(t) + D(t)u(t) \end{aligned} \tag{1.1}$$

---

[1] Dept. of Electrical and Computer Eng., 409 Dana Building, Northeastern University, Boston, MA 02115, Tel. (617) 437 5277

[2] Department of Electrical Eng., McGill University, Montréal, Canada H3A 2A7, Tel. (514) 398 7129

(As usual, we use the abbreviation $[A,\, B,\, C,\, D]$ in reference to (1.1).) Let $\Phi(t,s)$ be the fundamental matrix generated by A. (1.1) can be viewed as a realization of both a *causal input-output mapping (CI/O)* and of *anti-causal input-output mapping (ACI/O)*, as follows

$$CI/O:\; u(\cdot) \to y(\cdot) = D(\cdot)u(\cdot) + C(\cdot) \int_{-\infty}^{\cdot} \Phi(\cdot, s)B(s)u(s)ds \;:\; L_2 \to L_2 \qquad (1.2a)$$

$$ACI/O:\; u(\cdot) \to y(\cdot) = D(\cdot)u(\cdot) - C(\cdot) \int_{\cdot}^{\infty} \Phi(\cdot, s)B(s)u(s)ds \;:\; L_2 \to L_2 \qquad (1.2b)$$

A (possibly time-varying) direct-sum state-space partition is invariant when there holds

$$\Re^n = X_1(t) \oplus X_2(t), \quad \Phi(t,s)X_i(s) = X_i(t), \;\; i = 1, 2 \qquad (1.3)$$

We call (1.3) a *uniform partition* when the coordinate mapping $\mathcal{M}(t) : \xi \leftrightarrow (\xi_1, \xi_2) : \Re^n \leftrightarrow X_1(t) \times X_2(t)$ (where $\xi = \xi_1 + \xi_2$) is bounded both ways, uniformly for all t. If (1.3) is a uniform partition it associates (1.1) with a bounded coefficients block structure, with $\Phi = diag(\Phi_1, \Phi_2)$. Now (1.1) realizes also a mixed, non causal input-output operator

$$\begin{aligned} u(\cdot) \to y(\cdot) = D(\cdot)u(\cdot) + C_1(\cdot) &\int_{-\infty}^{\cdot} \Phi_1(\cdot, s)B_1(s)u(s)ds \\ - C_2(\cdot) &\int_{\cdot}^{\infty} \Phi_2(\cdot, s)B_2(s)u(s)ds \;:\; L_2 \to L_2 \end{aligned} \qquad (1.4)$$

In what follows we use the term *stability* in reference to uniform exponential stability: A is *stable* if for some positive M and a there holds $\|\Phi(t,s)\| \leq Me^{a(s-t)}, \quad t \geq s$. Accordingly, our system is *stabilizable* (resp. *detectable)* if $A - BK$ (resp. $A - LC$) is stable, for some uniformly bounded K (resp.L).

In the LTI case, the property of split stable — anti-stable dynamic (= no purely imaginary poles) is of prime importance (in particular, in Nehari Theorem related analysis). Among other implications, an associated state-space partition induces a unique **bounded**, usually mixed causal - anti-causal, I/O operator of the form (1.4). The following result characterizes such partitions in LTV systems.

Theorem I. 1. There is at most one stable — anti-stable invariant partition associated with a generator A.

2. Suppose (1.3) is an invariant uniform partition and that both the X_1 and the X_2 components of the evolution are uniformly controllable and observable. Then (1.4) defines a bounded operator iff (1.3) is a stable — anti-stable partition.

3. Suppose (1.3) is an invariant uniform partition and that both the X_1 and the X_2 components of the evolution are uniformly controllable and observable. Then (1.4) defines a bounded operator iff

$$\Re^n = X_2(t)^\perp \oplus X_1(t)^\perp \qquad (1.5)$$

defines a stable —anti-stable invariant uniform partition for the adjoint evolution.

4. An invariant, stable — anti-stable uniform partition (1.3) exists with the X_2 (= the anti-stable) component of the system being uniformly controllable and observable iff there exist uniformly bounded positive semidefinite solutions to the following two Riccati equations

$$\dot{P} + A'P + PA - PBB'P = 0 \tag{1.6a}$$

$$\dot{Q} - AQ - QA' + QC'CQ = 0 \tag{1.6b}$$

such that the generalized inverses of P and Q are uniformly bounded and the perturbed generators $A_P \triangleq A - BB'P$ and $A_Q \triangleq A - QC'C$ are stable. In that case $X_1(t) = kerP(t)$ and $X_2(t) = ImQ(t)$.

5. An invariant, stable — anti-stable uniform partition (1.3) exists with the X_1 (= the stable) component of the system being uniformly controllable and observable iff there exist uniformly bounded positive semidefinite solutions to the following two Riccati equations

$$\dot{R} + A'R + RA + RBB'R = 0 \tag{1.7a}$$

$$\dot{S} - AS - SA' - SC'CS = 0 \tag{1.7b}$$

such that the generalized inverses of R and S are uniformly bounded and the perturbed generators $A_R \triangleq A + BB'R$ and $A_S \triangleq A + SC'C$ are anti-stable. In that case $X_1(t) = ImS(t)$ and $X_2(t) = kerR(t)$.

6. Solutions to the Riccati equations (1.6) and (1.7) with the properties specified above, should they exist, are unique.

Outline of main ideas in the Proof. 1. The existence of two distinct stable — anti-stable invariant partitions would imply the existence of nonzero vectors such that either the forward or the backward evolutions emanating from them sustain both exponential growth and decline, which is a contradiction.

2. The "if" claim is obvious. In the "only if" claim we assume that (1.4) defines a bounded operator $\mathcal{T} : L_2(\Re) \to L_2(\Re)$. In particular, the restrictions of $\mathcal{T}$ to inputs on negative rays and outputs on the complementing positive rays, and vice versa, are uniformly bounded. This fact, coupled with the uniform controllability and observability assumptions guarantee bounds of the form

$$\begin{aligned} &\forall s, \xi_1 \quad \|\Phi_1(\cdot, s)\xi_1\|_{L_2(s,\infty)} \leq N_1\|\xi_1\| \\ &\forall s, \xi_2 \quad \|\Phi_2(\cdot, s)\xi_2\|_{L_2(-\infty,s)} \leq N_2\|\xi_2\| \end{aligned} \tag{1.8}$$

It is a well known fact (Datko 1972) that this implies stability of Φ_1 and anti-stability of Φ_2.

3. It is a straightforward observation that (1.3) is an invariant uniform partition for $\Phi(t,s)$ iff (1.7) is an invariant uniform partition for $\Phi(s,t)'$. By symmetry, it thus suffices to show that if the former is a stable — anti-stable partition, so is the latter. The proof is based on the uniform two-ways boundedness of the change of variables mapping

$$\mathcal{M} : X_1(t) \oplus X_1(t)^{\perp} \to X_1(t) \oplus X_2(t) \tag{1.9a}$$

$$\mathcal{N} : X_2(t)^{\perp} \oplus X_2(t) \to X_1(t) \oplus X_2(t) \tag{1.9b}$$

4,6. "*If*". Following from the two Riccati equations (1.6) there holds $P(t)\Phi_P(t,s) = \Phi(s,t)'P(s)$ and $\Phi_Q(t,s)Q(s) = Q(t)\Phi(s,t)'$. The invariance statements follow thereof. They imply, in turn, that

$$\Phi(t,s)\Big|_{X_1(s)} = \Phi_P(t,s)\Big|_{X_1(s)} \tag{1.10}$$

whence the stability of Φ over X_1. Similarly, anti-stability of $-A_P'$ implies anti-stability of the adjoint evolution over $X_1^\perp$. Symmetric arguments apply to the orthogonal partition $\Re^n = X_2^\perp \oplus X_2$.

Using the stability, anti-stability and invariance properties of the various subspaces above, one establishes the direct-sum property $\Re^n = X_1 \oplus X_2$. The fact that the latter is a **uniform** partition is verified as a consequence of the uniform boundedness of the generalized inverses P^- and Q^- and , in turn the two way boundedness of the projection from $X_2^\perp$ onto X_1.

To prove the uniqueness of the said solution (when existing) of the Riccati equation (1.6a), one inspects the optimization problem

$$\dot{x} = Ax + Bu, \quad x(s) = \xi$$
$$min\left\{ \|u\|_{L_2(s,\infty)} : x(t) \underset{t\to\infty}{\longrightarrow} 0 \;\&\; x \in L_2(s,\infty) \right\} \tag{1.11}$$

One is able to show that the unique optimal control is given in state feedback, by $u^* = -B'Px^*$, or in open-loop form, by $u^*(r) = -B(r)'P(r)\Phi_P(r,s)\zeta$. One also shows that when restricted to stabilizing, bounded, positive semidefinite solutions, (1.6a) can be equivalently written as

$$\begin{aligned} P(s) &= \int_s^\infty \Phi_P(r,s)'P(r)B(r)B(r)'P(r)\Phi_P(r,s)dr \\ &= \int_s^\infty P(s)\Phi(s,r)B(r)B(r)'\Phi(s,r)'P(s)dr \end{aligned} \tag{1.12}$$

The first equality implies the uniqueness of P as a consequence of the uniqueness of optimal controls in (1.11); taking into account the uniform boundedness of the generalized inverse P^-, the second equality is used to establish the uniform controllability of the anti-stable component of the system. A similar argument applies to uniqueness of the said solution of (1.6b) and the uniform observability claim.

"Only if." This part is established constructively. Again, two-ways uniform boundedness of the respective changes of co-ordinates mappings is critical.

5,6. This claim is established by symmetry to 4,6. ☐

2 Inner Operators and Co-Prime Factorizations

We say that an I/O operator $\mathcal{T}$ is *inner* if $\mathcal{T}'\mathcal{T} = I$ and *co-inner* if $\mathcal{T}\mathcal{T}' = I$. The following proposition translates well known facts pertaining to inner/co-inner transfer functions, to the LTV setting.

Observation 2.1. 1. Suppose $\exists \left(D'D\right)^{-1} \in L_\infty(\Re)$, $\exists\ P \geq 0,\ \in L_\infty(\Re)$ s.t. $A_P \triangleq A - B\left(D'D\right)^{-1}\left(D'C + B'P\right)$ is stable and satisfies the differential Riccati equation

$$\begin{aligned} &\dot{P} + P\left(A - B\left(D'D\right)^{-1}D'C\right) + \left(A - B\left(D'D\right)^{-1}D'C\right)'P \\ &- PB\left(D'D\right)^{-1}B'P + C'\left(I - D\left(D'D\right)^{-1}D'\right)C = 0 \end{aligned} \tag{2.1}$$

Denote $E \triangleq \left(D'D\right)^{-\frac{1}{2}}$ and $K \triangleq \left(D'D\right)^{-1}\left(D'C + B'P\right)$. Then the CI/O mapping in $\mathcal{R} \triangleq [A_P,\ BE,\ C - DK,\ DE]$ is inner.

2. Suppose $\exists \left(DD'\right)^{-1} \in L_\infty(\Re)$, $\exists\ Q \geq 0,\ \in L_\infty(\Re)$ s.t. $A_Q \triangleq A - \left(BD' + QC'\right)\left(DD'\right)^{-1} C$ is stable and satisfies the differential Riccati equation

$$\begin{aligned} &\dot{Q} - \left(A - BD'\left(DD'\right)^{-1}C\right)Q - Q\left(A - BD'\left(DD'\right)^{-1}C\right)' \\ &\qquad + QC'\left(DD'\right)^{-1}CQ - B\left(I - D'\left(DD'\right)^{-1}D\right)B' = 0 \end{aligned} \tag{2.2}$$

Denote $F \triangleq \left(DD'\right)^{-\frac{1}{2}}$ and $L \triangleq \left(BD' + QC'\right)\left(DD'\right)^{-1}$. Then the CI/O mapping in $\mathcal{S} \triangleq [A_Q,\ B - LD,\ FC,\ FD]$ is co-inner.

Outline of The Proof. 1. Let x be the internal state in the causal realization of $\mathcal{R}$. Unchanged, it remains the internal state in the realization of the left inverse $\mathcal{R}_{\mathcal{L}}^{-1}$, whereas $q = Px$ is the internal state of the adjoint, $\mathcal{R}'$, when it is applied to the output of $\mathcal{R}$. Comparing the output equations in the realizations of $\mathcal{R}'$ and $\mathcal{R}_{\mathcal{L}}^{-1}$ one obtains $\mathcal{R}'\mathcal{R} = \mathcal{R}_{\mathcal{L}}^{-1}\mathcal{R} = I$, as claimed.

The proof of Part 2 is similar. □

A causal *right co-prime factorization (RCF)* (respectively, *left co-prime factorization (LCF)*) of a CI/O mapping $\mathcal{T}$ is defined in terms of CI/O $\mathcal{N}_{\mathcal{R}}$ and $\mathcal{D}_{\mathcal{R}}$ (respectively $\mathcal{N}_{\mathcal{L}}$ and $\mathcal{D}_{\mathcal{L}}$), of stable systems, such that $\mathcal{D}_{\mathcal{R}}$ (respectively $\mathcal{D}_{\mathcal{L}}$) has a causal inverse[3], there holds $\mathcal{T} = \mathcal{N}_{\mathcal{R}}\mathcal{D}_{\mathcal{R}}^{-1}$ (respectively, $\mathcal{T} = \mathcal{D}_{\mathcal{L}}^{-1}\mathcal{N}_{\mathcal{L}}$) and such that $\begin{bmatrix} \mathcal{N}_{\mathcal{R}} \\ \mathcal{D}_{\mathcal{R}} \end{bmatrix}$ has a stable CI/O left inverse (respectively, $[\mathcal{N}_{\mathcal{L}} \quad \mathcal{D}_{\mathcal{L}}]$ has a stable CI/O right inverse). These are *normalized co-prime factorizations (NRCF and NLCF)* if $\begin{bmatrix} \mathcal{N}_{\mathcal{R}} \\ \mathcal{D}_{\mathcal{R}} \end{bmatrix}$ is inner (respectively, $[\mathcal{N}_{\mathcal{L}} \quad \mathcal{D}_{\mathcal{L}}]$ is co-inner).

The following is a standard observation (see, e.g. Vidyasagar 1985) in the LTI case. It carries, essentially with no change in proof to the time varying setting.

Observation 2.2. If K and L are uniformly bounded and $A - BK$ and $A - LC$ are stable then $\mathcal{T}$, the CI/O of (1.1), has an RCF where $\mathcal{N}_{\mathcal{R}}$ and $\mathcal{D}_{\mathcal{R}}$ have the respective realizations $[A - BK, B, C - DK, D]$ and $[A - BK, B, -K, I]$ and an LCF, given with the realizations $[A - LC, B - LD, C, D]$ and $[A - LC, -L, C, I]$ for $\mathcal{N}_{\mathcal{L}}$ and $\mathcal{D}_{\mathcal{L}}$, respectively.

These next two corollaries extend known facts on co-prime factorization from the time-invariant to the time-varying case.

Corollary 2.3. Suppose that solutions to the Riccati equations (1.6) exist, as described in Theorem I.4. Then $\mathcal{T}$ has RCF and LCF with inner and co-inner denominators, respectively. Such factorizations are obtained setting $K \triangleq B'P$ and $L \triangleq QC'$ in the formulae suggested in Observation 2.2.

Proof. Substituting and $D = I$ and $C = 0$ brings (2.1) to the form of (1.6a). By Observation 2.1.1, the suggested $\mathcal{D}_{\mathcal{R}}$ is inner. Similarly, Observation 2.1.2 implies that $\mathcal{D}_{\mathcal{L}}$ is co-inner. □

This next corollary is not needed in the present discussion, and we bring it merely as an illustration of the scope of the time-domain LQ optimization approach.

---

[3] i.e. the D matrix in its realization is invertible.

Corollary 2.4. Suppose that (1.1) is uniformly stabilizable and detectable. Then T has NRCF and NLCF as follows. $\exists\ P, Q > 0$ s.t. $P, Q, P^{-1}, Q^{-1} \in L_\infty$, $A_K \triangleq A - BK$ and $A_L \triangleq A - LC$ are stable, where $K \triangleq \left(I + D'D\right)^{-1}\left(D'C + B'P\right)$ and $L \triangleq \left(BD' + QC'\right)\left(I + DD'\right)^{-1}$, and the following Riccati equations are satisfied

$$\begin{aligned}\dot{P} + P\left(A - B\left(I + D'D\right)^{-1}D'C\right) + \left(A - B\left(I + D'D\right)^{-1}D'C\right)'P \\ - PB\left(I + D'D\right)^{-1}B'P + C'\left(I + DD'\right)^{-1}C = 0\end{aligned} \tag{2.3a}$$

$$\begin{aligned}\dot{Q} - \left(A - BD'\left(I + DD'\right)^{-1}C\right)Q - Q\left(A - BD'\left(I + DD'\right)^{-1}C\right)' \\ + QC'\left(I + DD'\right)^{-1}CQ - B\left(I + D'D\right)^{-1}B' = 0\end{aligned} \tag{2.3b}$$

Define $E \triangleq \left(I + D'D\right)^{-\frac{1}{2}}$ and $F \triangleq \left(I + DD'\right)^{-\frac{1}{2}}$. Let $\mathcal{N}_\mathcal{R}, \mathcal{D}_\mathcal{R}, \mathcal{N}_\mathcal{L}$ and $\mathcal{D}_\mathcal{L}$ be as defined in observation 2.2, and set $\widetilde{\mathcal{N}_\mathcal{R}} \triangleq \mathcal{N}_\mathcal{R}E$, $\widetilde{\mathcal{D}_\mathcal{R}} \triangleq \mathcal{D}_\mathcal{R}E$, $\widetilde{\mathcal{N}_\mathcal{L}} \triangleq F\mathcal{N}_\mathcal{L}$ and $\widetilde{\mathcal{D}_\mathcal{L}} \triangleq F\mathcal{D}_\mathcal{L}$. The normalized co-prime factorizations are given by the two pairs $\widetilde{\mathcal{N}_\mathcal{R}}$, $\widetilde{\mathcal{D}_\mathcal{R}}$, $\widetilde{\mathcal{N}_\mathcal{L}}$ and $\widetilde{\mathcal{D}_\mathcal{L}}$.

Outline of The Proof. Existence and uniqueness of the said solution to the Riccati equation (2.3a) follows from the existence and uniqueness of a control minimizing the nonsingular cost index $J(\xi, s, \infty) \triangleq \|y\|_{L_2}^2 + \|u\|_{L_2}^2$, subject to the dynamics of (1.1). Under a stabilizability and detectability assumption that is a standard result and the optimal control is given in stabilizing feedback form, by $u = -Kx$. A similar existence and uniqueness argument applies to (2.3b).

Substitute $\begin{bmatrix} D \\ I \end{bmatrix}$ for D and $\begin{bmatrix} C \\ 0 \end{bmatrix}$ for C in the statement of Observation 2.1.1, and $[D \quad I]$ for D and $[B \quad 0]$ for B in Observation 2.1.2, to deduce that $\begin{bmatrix} \widetilde{\mathcal{N}_\mathcal{R}} \\ \widetilde{\mathcal{D}_\mathcal{R}} \end{bmatrix}$ is inner and $[\widetilde{\mathcal{N}_\mathcal{L}} \quad \widetilde{\mathcal{D}_\mathcal{L}}]$ is co-inner. □

Remark. It is very useful to think of the following as an independent source of intuition, leading to the two Corollaries: It is a standard observation in the context of an LQ optimization problem that the mapping from the momentary deviation from feedback input optimality to the controlled variable (= the variable which L_2 norm is to be minimized) defines a weighted L_2 isometry. In the context of Corollary 2.3, the Riccati equations solve minimum input energy problems. Thus the controlled variable is the control input itself. Adding the weights, the right and left "denominators" are then the "error mapping". As isometries, they define inner/co-inner mappings. In Corollary 2.4 the controlled variable is the point in the system's graph, associated with a specified input. The "error mapping" thus defines a weighted isometry into the graph and, in turn, a NRCF or NLCF.

3 Nehari Theorem

In the LTI case, when the transfer function associated with (1.1), $T(s)$, belongs to $L_\infty(j\Re)$, *Nehari theorem* evaluates its $L_\infty(j\Re)$ distance from the stable transfer functions

$$\inf_{\Theta \in H_\infty} \|T - \Theta\|_{L_\infty(j\Re)} \tag{3.1}$$

The relevance of the $L_\infty(j\Re)$ norm stems from the fact that it is equal to the $L_2(j\Re)$—induced norm of the associated multiplication operator.

Assume that solutions to the two Riccati equations (1.6) exist and let (1.3) be the associated stable — anti-stable uniform state space partition, as described in Theorem I.4. We then suggest the following as a time-domain interpretation of (3.1). Let $\mathcal{T}$ be the bounded, causal — anti-causal I/O operator of the form (1.4), as associated with (1.3). We look for the minimum $L_2(\Re)$—induced norm distance between $\mathcal{T}$ and the class of bounded causal I/O operators.

The following observation translates the LTV Nehari problem into the familiar one block model matching form.

Observation 3.1. 1. Let $\mathcal{N_L}$ and $\mathcal{D_L}$ be the LCF suggested in Corollary 2.3. Then $\mathcal{T} = \mathcal{D}_\mathcal{L}^{'}\mathcal{N_L}$. 2. For any bounded I/O operator Θ there holds

$$\|\mathcal{T} - \Theta\| = \|\mathcal{N_L} - \mathcal{D_L}\Theta\| \tag{3.2}$$

Proof. Part 1 follows from direct analysis of state-space representations of the various operators involved, and from the fact that with each state-space representation with a stable — anti-stable state-space partition there is associated a unique bounded I/O operator of the form (1.4). It also easily follows that our $\mathcal{D_L}$ is **both** inner and co-inner. Part 2 is a consequence of the latter fact. ☐

Theorem II (Nehari). Let

$$\gamma_0 = \underset{\substack{\Theta \text{ is a bounded} \\ {}_{CI/O}\text{operator}}}{inf} \|\mathcal{N_L} - \mathcal{D_L}\Theta\| \tag{3.3}$$

Then

$$\gamma_0^2 = \underset{t}{max}\, \sigma_{max}\Big((Q^-)^{\frac{1}{2}} P^- (Q^-)^{\frac{1}{2}}\Big)(t) \tag{3.4}$$

Outline of The Proof. Using the explicit state-space formulae for $\mathcal{N_L}$ and $\mathcal{D_L}$, as well as the stable — anti-stable state-space partition in (1.1), one shows that there exists a bounded CI/O Θ s.t. $\|\mathcal{N_L} - \mathcal{D_L}\Theta\| < \gamma$ if and only if there exists a stabilizing feedback policy[4] $u = Kx_2$ in the system

$$\dot{x}_2 = A_2 x - Q_2 C_2^{'} u + B_2 w \tag{3.5}$$

s.t. the closed-loop mapping $\Xi : w \rightarrow u$ satisfies the induced norm bound $\|\Xi\| < \gamma$.

Denote

$$R(t) \triangleq \int_t^\infty \Phi_2(t,s)\Big(Q_2 C_2^{'} C_2 Q_2\Big)(s)\Phi_2(t,s)^{'} ds \tag{3.6}$$

Using our assumptions on Q one shows that R is a uniformly positive, uniformly bounded matrix function. Moreover, given any initial time and initial state, the following open-loop input policy in (3.5)

$$u(r) = (C_2 Q_2)(r)\Phi_2(t,r)^{'} R(t)^{-1}\left(\int_t^\infty \Phi_2(t,s)B_2(s)w(s)ds + x_2(t)\right), \quad r \geq t \tag{3.7}$$

---

[4] Here $x_2(t) \in X_2(t)$ and the other coefficients are the associated X_2 component of the system. In particular, Q_2 is the uniformly positive $X_2 \times X_2$ component of Q.

(denoted in short $u = \mathcal{U}(t, x_2(t), w)$) provides the unique optimal solution to the minimum energy optimal control problem

$$\min_{u} \left\{ \|u\|_{L_2(t,\infty)} : \lim_{s \to \infty} x_2(s) = 0 \ \& \ x_2 \in L_2(t,\infty) \right\} \tag{3.8}$$

As the open-loop optimal control is certainly not inferior to any feedback policy, if indeed there exists a stabilizing feedback policy in (3.5) s.t. $\|\Xi\| < \gamma$ then there is some positive δ s.t.

$$\forall t, w \quad \gamma^2 \|w\|^2 - \|\mathcal{U}(t,0,w)\|^2 \geq \delta^2 \|w\|^2 \tag{3.9}$$

Following from analysis which is completely similar to that of Tadmor 1989a,b, 1990a, one then shows that there then exists a unique minimizing w in the optimization problem

$$\alpha_{min}(t, x_2(t)) = \min_{w} \left\{ \gamma^2 \|w\|^2 - \|\mathcal{U}(t, x_2(t), w)\|^2 \right\} \tag{3.10}$$

The optimal solution is associated with a uniformly bounded and positive matrix function Π_2 s.t. $(A_2)_{\Pi_2} \triangleq A_2 + \left(\frac{1}{\gamma^2} B_2 B_2' - Q_2 C_2' C_2 Q_2\right)\Pi_2$ is stable and

$$\dot{\Pi}_2 + \Pi_2 A_2 + A_2' \Pi_2 + \Pi_2 \left(\frac{1}{\gamma^2} B_2 B_2' - Q_2 C_2' C_2 Q_2 \right) \Pi_2 = 0 \tag{3.11}$$

It can be further shown that

$$\begin{aligned} \Pi_2(t) &= \left(Q_2(t) - \frac{1}{\gamma^2} \int_t^\infty \Phi_2(t,r) \left(B_2 B_2' \right)(r) \Phi_2(t,r)' \, dr \right) \\ &= Q_2(t)^{-\frac{1}{2}} \left(I - Q_2(t)^{-\frac{1}{2}} \frac{1}{\gamma^2} \int_t^\infty \Phi_2(t,r) \left(B_2 B_2' \right)(r) \Phi_2(t,r)' \, dr Q_2(t)^{-\frac{1}{2}} \right) Q_2(t)^{-\frac{1}{2}} \end{aligned} \tag{3.12}$$

and that the integral

$$\int_t^\infty \Phi_2(t,r) \left(B_2 B_2' \right)(r) \Phi_2(t,r)' \, dr \tag{3.13}$$

provides the compression of the generalized inverse $P(t)^-$ to the subspace $X_2(t)$. The uniform positivity of Π_2 implies therefore uniform positivity of $\gamma^2 I - (Q^-)^{\frac{1}{2}} P^- (Q^-)^{\frac{1}{2}}$ which complete the proof of necessity in the Theorem.

Conversely, suppose $\gamma^2 I - (Q^-)^{\frac{1}{2}} P^- (Q^-)^{\frac{1}{2}}$ is uniformly positive. Then (3.12) defines a uniformly positive definite, uniformly bounded solution of (3.11). Let $\Omega_2(t,s)$ be the transition generated by $(A_2)_{\Pi_2}$. It is easy to deduce from the (3.11) that there holds $\Omega_2(t,s) = \Pi_2(t)^{-1} \Phi_2(s,t)' \Pi_2(s)$. The stability of $(A_2)_{\Pi_2}$ is thus a consequence of the anti-stability of A_2.

Having that established one traces standard state-feedback H_∞ optimization analysis (e.g. the proof of sufficiency in Tadmor 1989b, Theorem II), to deduce that the state feedback $u = C_2 Q_2 \Pi_2 x_2$ is stabilizing and strictly γ suboptimal in (3.5). □

Note. The LTV system (1.1) is associated with a parametrized family of Hankel operators

$$(\Gamma_{\mathcal{T}}(s)u)(t) = \begin{cases} 0 & t \geq s \\ C_2(t) \int_s^{\infty} \Phi_2(t,r) B_2(r) u(r) dr & t < s \end{cases} \tag{3.14}$$

It represents the effects of "future" inputs on "past" outputs in the anti-causal component of the operator $\mathcal{T}$. It easily follows that the induced norm of $\Gamma_{\mathcal{T}}(s)$ is given by

$$\|\Gamma_{\mathcal{T}}(s)\|^2 = \sigma_{max}(Q^-)^{\frac{1}{2}} P^- (Q^-)^{\frac{1}{2}}(s) \tag{3.15}$$

Thus, as in the LTI case, Nehari Theorem states that the distance between a bounded, non-causal I/O operator $\mathcal{T}$ and the family of CI/O mappings in stable systems is the supremal induced norm of the family of Hankel operators generated by $\mathcal{T}$.

We complete the discussion with the following.

Corollary 3.1. Given $\gamma > \gamma_0$, the family of causal stable systems $\Theta : w \to u$ s.t. $\|\mathcal{N}_{\mathcal{L}} - \mathcal{D}_{\mathcal{L}}\Theta\| < \gamma$ is parametrized as follows. Let Π be the matrix function which $X_2 \times X_2$ block is Π_2, as above, complemented by three zero blocks to an $n \times n$ matrix. Then set

$$\begin{aligned} \dot{p} &= \left(A - QC'CQ\Pi\right)p + Bw - QC'v \\ q &= -\frac{1}{\gamma^2}B'\Pi x \quad w, \qquad\qquad v = \Xi q \\ u &= C(I - Q\Pi)p + Dw - v \end{aligned} \tag{3.16}$$

where Ξ is allowed to be any causal stable system, subject to the induced norm bound $\|\Xi\| < \gamma$.

Outline of The Main Ideas in The Proof. In the proof of necessity, the basic observation is, as in previous parts of the discussion, that the mapping from the momentary deviation from feedback optimal input in the combined min-max problem (3.8)-(3.10), to the controlled variable u, is a weighted isometry; here the weight being a scalar smaller than γ. Denoting that mapping by Ξ, it follows from direct computation that Θ must be of the form (3.16). In the proof of sufficiency, the same argument guarantees that (3.16) entails strict γ suboptimality. The proof of internal stability in (3.16) is similar to counterpart arguments in Tadmor 1989a, 1990a. ☐

References

1. V.M. Adamjan, D.Z Arov & M.G. Krein 1971, "*Analytic properties of Schmidt pairs for a Hankel operator and the generalized Schur-Takagi problem*", Math. USSR Sbornik 15, 31-73
2. J. A. Ball & J. W. Helton 1983, "*A Beurling-Lax Theorem for the Lie group U(m,n) which contains most classical interpolation theory*", J. Operator Theo. 9, 107-142
3. J. A. Ball & A. C. M. Ran 1987, "*Optimal Hankel norm model reduction and Wiener-Hopf factorization I: The Canonical case*", SIAM J. Control & Optim 25
4. J. A. Ball, I. Gohberg & L. Rodman 1988, "*Realization and interpolation of rational matrix functions*" in **Operator Theory, Advances and Applications** 33, I. Gohberg Ed., Birkhauser

5. Dym, H. 1989, J Contractive Matrix Functions, Reproducing Kernels, Hilbert Spaces and Interpolation, Conference Board of The Math. Sciences #71, AMS
6. A. Feintuch & B. A. Francis 1985, "*Uniformly optimal control of linear systems*", Automatica 21, 563–574
7. B. A. Francis 1987, **A Course In $\mathbf{H}_\infty$ Control**, Springer
8. K. Glover, "*Robust stabilization of linear multivariable systems: relations to approximations*", Int. J. Control 43, 741–76
9. I. Gohberg, M. A. Kasshoek & H. J. Woerdman 1989a, "*The band method for positive and contractive extension problem*", J. Operator Theory 22, 109–155; 1989b, "*The band method for positive and contractive extension problem: An alternative method and new applications*", Integral Eqs. & Operator Theory 12, 343–382
10. Z. Nehari, 1957, "*On bounded bilinear forms*", Ann. of Math. 65, 153-162
11. S.C. Power 1982, **Hankel Operators on Hilbert Spaces**, Pitman
12. D. Sarason 1967, "*Generalize interpolation in H_∞*", Trans. AMS 127, pp. 179-203
13. G. Tadmor 1988, "*H_∞ in the time domain: The standard four block problem*" LCDS/CCS Report #88-21, Brown University; 1989a, "*Time domain optimal control and worst case linear systems design*", Proc. of 28th CDC; 1989b, "*The standard H_∞ problem and the maximum principle: The general linear case*", UTD Technical Report #189, The University of Texas, Dallas (to appear in SIAM J. Control & Optim; 1990a, "*Worst case design in the time domain: The maximum principle and the standard H_∞ problem*", MCSS 3, 301-324; 1990b, "*The Input/output norms in general linear systems*", Int. J. Control 51, 911-924
14. M. S. Verma 1989, "*Robust stabilization of linear time invariant plants*", IEEE Trans Aut. Control AC-34, 870–875.
15. M. Vidyasagar 1985, **Control Systems Synthesis: A Factorization Approach**, MIT Press

Robustness analysis and optimization.

Jacob Kogan
Department of Mathematics
University of Maryland Baltimore County
Baltimore, Maryland 21228

Abstract. The paper highlights an optimization approach to problems of robustness in the parameter space. Specifically we apply optimization tools to problems of robust stability and robust positivity of families of polynomials and derive necessary and sufficient robustness conditions for these problems.

1. Introduction

The purpose of this paper is to advocate an optimization approach to problems of robustness. As an example we consider three specific problems:

1. l^p robust Hurwitz stability of families of polynomials.
2. l^∞ robust Schur stability of families of polynomials.
3. l^p robust positivity of polynomials with real coefficients.

The optimization approach provides necessary and sufficient conditions for robustness in all three cases. We present a detailed solution for the first problem, and state main results for two others. It is hoped that applications of optimization techniques could be helpful for investigations of general robustness problems.

2. l^p robust Hurwitz stability of a family of polynomials

Let a nominal polynomial $\mathbf{p}^0(z) = (a_0^0 + jb_0^0) + (a_1^0 + jb_1^0)z + \ldots + (a_n^0 + jb_n^0)z^n$ of degree n, $2n+2$ positive numbers α_k, β_k, $k = 0, 1, \ldots, n$, and $p \in [1, \infty]$ be given. For $\gamma \geq 0$ we consider a family of polynomials $\mathcal{P}_\gamma$ defined as follows:

$$\mathcal{P}_\gamma = \left\{ \mathbf{p}(z) \; : \; \sum_k \left| \frac{a_k^0 - a_k}{\alpha_k} \right|^p + \sum_k \left| \frac{b_k^0 - b_k}{\beta_k} \right|^p \leq \gamma^p \right\}. \tag{2.1}$$

In the sequel we denote $\left[\sum_k \left| \frac{a_k^0 - a_k}{\alpha_k} \right|^p + \sum_k \left| \frac{b_k^0 - b_k}{\beta_k} \right|^p \right]^{1/p}$ by $||\mathbf{p}^0(z) - \mathbf{p}(z)||_p$.

We say that a polynomial $\mathbf{p}(z)$ is Hurwitz if $\sigma < 0$ whenever $\mathbf{p}(\sigma + jw) = 0$. A set of polynomials is said to be Hurwitz if and only if every element of the set is Hurwitz. We assume that the nominal polynomial $\mathbf{p}^0(z)$ is stable and define its stability radius $r_H(p, \boldsymbol{\alpha}, \boldsymbol{\beta})$ as follows

$$r_H(p, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \inf\{||\mathbf{p}^0(z) - \mathbf{p}(z)||_p, \; \mathbf{p}(z) \text{ fails to be Hurwitz}\}. \tag{2.2}$$

A way to determine the stability radius is the following: Denote by $\mathcal{Z}_\gamma$ the set of all roots of the family $\mathcal{P}_\gamma$ (so, for example, $\mathcal{Z}_0$ is the set of all roots of the nominal polynomial $\mathbf{p}^0(z)$). The set $\mathcal{Z}_0$ lies in the left–half plane, when γ is small the "cloud" of roots $\mathcal{Z}_\gamma$ is located in the left–half plane. As γ grows $\mathcal{Z}_\gamma$ grows until it touches the imaginary axis (the boundary of the stability region). The minimal γ whose "cloud" of roots $\mathcal{Z}_\gamma$ touches the imaginary axis determines $r_H(p, \boldsymbol{\alpha}, \boldsymbol{\beta})$. Since the "contact" point jw is unknown we first try to evaluate

$$r_1(w) = \inf\{||\mathbf{p}^0(z) - \mathbf{p}(z)||_p,\ \mathbf{p}(jw) = 0\}. \tag{2.3}$$

for each fixed frequency w. Next a "brute force search" over the frequency domain is carried out, i.e., $r_H(p, \boldsymbol{\alpha}, \boldsymbol{\beta}) = r_1 = \inf r_1(w)$.

The arguments can be traced back to the work of Frazer and Duncan [FD], where a family of real polynomials ($b_i = 0,\ i = 0, \ldots, n$) with positive leading coefficients ($a_n > 0$) is considered. However, when one deals with polynomials without additional degree assumptions the "degree dropping" provides an additional reason for the lost of stability, and

$$r_H(p, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \min\{r_1,\ r_2\},\ \text{ where } r_2 = \left(\left|\frac{a_n^0}{\alpha_n}\right|^p + \left|\frac{b_n^0}{\beta_n}\right|^p\right)^{1/p}. \tag{2.4}$$

Since the evaluation of r_2 is straightforward, we concentrate on solving (2.3). We consider first the case of $1 < p < \infty$. For a polynomial $\mathbf{p}(z)$ we denote $\mathbf{p}(jw)$ by $U(w) + jV(w)$ where

$$U(w) = a_0 - b_1 w - a_2 w^2 + b_3 w^3 + \ldots = \sum_{k=0}^{n} u_k w^k,$$

$$V(w) = b_0 + a_1 w - b_2 w^2 - a_3 w^3 + \ldots = \sum_{k=0}^{n} v_k w^k.$$

The condition $0 = \mathbf{p}(jw) = U(w) + jV(w)$ implies $U(w) = V(w) = 0$, hence

$$U(w) - U^0(w) = -U^0(w),\ \text{and}\ V(w) - V^0(w) = -V^0(w),$$

here $U^0(w)$ and $V^0(w)$ are real and imaginary parts of $\mathbf{p}^0(jw)$. We denote

$$\sum_k \left|\frac{u_k - u_k^0}{A_k}\right|^p + \sum_k \left|\frac{v_k - v_k^0}{B_k}\right|^p \quad \text{by } f(\mathbf{u}, \mathbf{v}), \tag{2.5}$$

where

$$\begin{aligned} A_{2i} &= \alpha_{2i}, \quad A_{2i+1} = \beta_{2i+1}, \\ B_{2i} &= \beta_{2i}, \quad B_{2i+1} = \alpha_{2i+1}, \end{aligned} \tag{2.6}$$

and rewrite (2.3) as follows:

$$\min f(\mathbf{u}, \mathbf{v}) \tag{2.7}$$

subject to two linear constraints

$$\sum_k (u_k - u_k^0) w^k = -\sum_k u_k^0 w^k \tag{2.8}$$

$$\sum_k (v_k - v_k^0) w^k = -\sum_k v_k^0 w^k \tag{2.9}$$

The constrained optimization problem can be reduced to the unconstrained problem:

$$\sup_{\lambda,\mu} \inf_{\mathbf{u},\mathbf{v}} F(\mathbf{u},\mathbf{v},\lambda,\mu) \tag{2.10}$$

where

$$F(\mathbf{u},\mathbf{v},\lambda,\mu) = f(\mathbf{u},\mathbf{v}) + \lambda\left\{\sum_k (u_k - u_k^0) w^k + \sum_k u_k^0 w^k\right\} + \mu\left\{\sum_k (v_k - v_k^0) w^k + \sum_k v_k^0 w^k\right\},$$

and λ, μ are the Lagrange multipliers (see e.g., [R]). The optimality conditions are given by the next four equations:

$$0 = \frac{\partial F}{\partial u_i} = \frac{p}{A_k}\left|\frac{u_k - u_k^0}{A_k}\right|^{p-1} \operatorname{sgn}(u_k - u_k^0) + \lambda w^k, \qquad k = 0, \ldots, n \tag{2.11a}$$

$$0 = \frac{\partial F}{\partial v_i} = \frac{p}{B_k}\left|\frac{v_k - v_k^0}{B_k}\right|^{p-1} \operatorname{sgn}(v_k - v_k^0) + \mu w^k, \qquad k = 0, \ldots, n. \tag{2.11b}$$

$$0 = \frac{\partial F}{\partial \lambda} = \sum_k (u_k - u_k^0) w^k + U^0(w), \tag{2.11c}$$

$$0 = \frac{\partial F}{\partial \mu} = \sum_k (v_k - v_k^0) w^k + V^0(w). \tag{2.11d}$$

Equations (2.11a), (2.11b) in particular imply

$$\operatorname{sgn}(u_k - u_k^0) = \operatorname{sgn}(-\lambda w^k), \text{ and } \operatorname{sgn}(v_k - v_k^0) = \operatorname{sgn}(-\mu w^k),\ k = 0, \ldots, n. \tag{2.12}$$

Note that $\alpha|x|^{p-1}\operatorname{sgn}(x) = \beta$ if and only if $x = \left|\frac{\beta}{\alpha}\right|^{q/p} \operatorname{sgn}(x)$, where $1/p + 1/q = 1$. This observation combined with (2.11a) and (2.11b) yields:

$$u_k - u_k^0 = p^{-q/p}|\lambda|^{q/p}|w|^{kq/p} A_k^q \operatorname{sgn}(u_k - u_k^0), \tag{2.13a}$$

$$v_k - v_k^0 = p^{-q/p}|\mu|^{q/p}|w|^{kq/p} B_k^q \operatorname{sgn}(v_k - v_k^0). \tag{2.13b}$$

Due to (2.11c) and (2.11d) one has

$$\operatorname{sgn}(-\lambda) p^{-q/p}|\lambda|^{q/p} \sum_k A_k^q |w|^{kq} = -U^0(w), \tag{2.13c}$$

$$\operatorname{sgn}(-\mu) p^{-q/p}|\mu|^{q/p} \sum_k B_k^q |w|^{kq} = -V^0(w). \tag{2.13d}$$

Hence

$$|\lambda|^{q/p} = -\frac{p^{q/p} U^0(w) \operatorname{sgn}(-\lambda)}{\sum_k A_k^q |w|^{kq}}, \text{ and } |\mu|^{q/p} = -\frac{p^{q/p} V^0(w) \operatorname{sgn}(-\mu)}{\sum_k B_k^q |w|^{kq}}. \tag{2.14}$$

Now one can write

$$u_k - u_k^0 = -\frac{U^0(w)\operatorname{sgn}(-\lambda)}{\sum_l A_l^q |w|^{lq}} A_k^q |w|^{kq/p} \operatorname{sgn}(u_k - u_k^0), \tag{2.15a}$$

and

$$v_k - v_k^0 = -\frac{V^0(w)\mathrm{sgn}(-\mu)}{\sum_l A_l^q |w|^{lq}} A_k^q |w|^{kq/p} \mathrm{sgn}(v_k - v_k^0). \tag{2.15b}$$

So that

$$\left|\frac{u_k - u_k^0}{A_k}\right|^p = |U^0(w)|^p \left|\frac{A_k^q |w|^{kq/p}}{A_k \sum_l A_l^q |w|^{lq}}\right|^p = |U^0(w)|^p \left|\frac{A_k^{q/p} |w|^{kq/p}}{\sum_l A_l^q |w|^{lq}}\right|^p$$

$$\left|\frac{v_k - v_k^0}{B_k}\right|^p = |V^0(w)|^p \left|\frac{B_k^q |w|^{kq/p}}{B_k \sum_l B_l^q |w|^{lq}}\right|^p = |V^0(w)|^p \left|\frac{B_k^{q/p} |w|^{kq/p}}{\sum_l B_l^q |w|^{lq}}\right|^p$$

Finally

$$r_1^p(w) = \sum_k \left|\frac{u_k - u_k^0}{A_k}\right|^p + \sum_k \left|\frac{v_k - v_k^0}{B_k}\right|^p =$$

$$\frac{|U^0(w)|^p}{|\sum_k A_k^q |w|^{kq}|^p} \left\{\sum_k A_k^q |w|^{kq}\right\} + \frac{|V^0(w)|^p}{|\sum_k B_k^q |w|^{kq}|^p} \left\{\sum_k B_k^q |w|^{kq}\right\} =$$

$$\left|\frac{U^0(w)}{\{\sum_k A_k^q |w|^{kq}\}^{1/q}}\right|^p + \left|\frac{V^0(w)}{\{\sum_k B_k^q |w|^{kq}\}^{1/q}}\right|^p.$$

In order to express r_1 in a convenient form we introduce two auxiliary functions

$$S_q(w) = \{|\alpha_0|^q + |\beta_1 w|^q + |\alpha_2 w^2|^q + |\beta_3 w^3|^q + \ldots\}^{1/q}, \tag{2.16}$$

$$T_q(w) = \{|\beta_0|^q + |\alpha_1 w|^q + |\beta_2 w^2|^q + |\alpha_3 w^3|^q + \ldots\}^{1/q}, \tag{2.17}$$

Keeping in mind (2.4) one obtains the following expression for r_1

$$r_1 = \inf_w \left[\left|\frac{U^0(w)}{S_q(w)}\right|^p + \left|\frac{V^0(w)}{T_q(w)}\right|^p\right]^{\frac{1}{p}}.$$

The formula (2.4) then becomes

$$r_H(p, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \min\left\{r_1, \left(\left|\frac{a_n^0}{\alpha_n}\right|^p + \left|\frac{b_n^0}{\beta_n}\right|^p\right)^{1/p}\right\}. \tag{2.18}$$

A simple continuity argument shows that (2.18) holds for any $1 \le p \le \infty$. The robust stability conditions are given next.

Theorem 2.1 Suppose that the nominal polynomial $p^0(z)$ is Hurwitz.

1. If $\gamma < r_H(p, \boldsymbol{\alpha}, \boldsymbol{\beta})$ the family of polynomials $\mathcal{P}_\gamma$ is Hurwitz.
2. If $\gamma > r_H(p, \boldsymbol{\alpha}, \boldsymbol{\beta})$ the family of polynomials $\mathcal{P}_\gamma$ fails to be Hurwitz.

Finally, we note that

$$r_1 = \inf_w \left[\left|\frac{U^0(w)}{S_q(w)}\right|^p + \left|\frac{V^0(w)}{T_q(w)}\right|^p\right]^{\frac{1}{p}} \le \lim_{w\to\infty} \left[\left|\frac{U^0(w)}{S_q(w)}\right|^p + \left|\frac{V^0(w)}{T_q(w)}\right|^p\right]^{\frac{1}{p}} = \left(\left|\frac{a_n^0}{\alpha_n}\right|^p + \left|\frac{b_n^0}{\beta_n}\right|^p\right)^{1/p} = r_2, \tag{2.19}$$

and for families of polynomials with complex coefficients $r_H(p, \boldsymbol{\alpha}, \boldsymbol{\beta})$ is always determined by r_1.

When the nominal polynomial $\mathbf{p}^0(z) = a_0^0 + a_1^0 z + \ldots + a_n^0 z^n$ has real coefficients only, one can obtain necessary and sufficient conditions for l^p Hurwitz stability of polynomials with real coefficients by setting $\boldsymbol{\beta} = \mathbf{0}$. The family of polynomials $\mathcal{P}_\gamma^r$ then is given by:

$$\mathcal{P}_\gamma^r = \left\{ \mathbf{p}(z) \, : \, \sum_k \left| \frac{a_k^0 - a_k}{\alpha_k} \right|^p \leq \gamma^p \right\}. \tag{2.20}$$

The functions $T_q(w)$ and $S_q(w)$ should be modified as follows:

$$T_q^r(w) = \{ |\alpha_1 w|^q + |\alpha_3 w^3|^q + |\alpha_5 w^5|^q + \ldots \}^{1/q}, \tag{2.21}$$

$$S_q^r(w) = \{ |\alpha_0|^q + |\alpha_2 w^2|^q + |\alpha_4 w^4|^q + |\alpha_6 w^6|^q + \ldots \}^{1/q}. \tag{2.22}$$

Let $r_H(p, \boldsymbol{\alpha}) = \min \left\{ \inf_w \left[\left| \frac{V^0(w)}{T_q^r(w)} \right|^p + \left| \frac{U^0(w)}{S_q^r(w)} \right|^p \right]^{\frac{1}{p}}, \left| \frac{a_n^0}{\alpha_n} \right| \right\}$. The robust stability conditions then become:

Theorem 2.2 Suppose that the nominal polynomial $\mathbf{p}^0(z)$ is stable.

1. If $\gamma < r_H(p, \boldsymbol{\alpha})$ then the family of polynomials $\mathcal{P}_\gamma^r$ is Hurwitz.
2. If $\gamma > r_H(p, \boldsymbol{\alpha})$ then the family of polynomials $\mathcal{P}_\gamma^r$ fails to be Hurwitz.

While the stability radius for a family of complex polynomials does not depend on r_2 (see (2.19)), as we show by an example in the case of polynomials with real coefficients r_2 can not be ignored.

Example 2.1. Consider the nominal polynomial

$$\mathbf{p}^0(z) = 2 + 2z + z^2 = (z + 1 + j)(z + 1 - j).$$

Let $\alpha_0 = \alpha_1 = \alpha_2 = 1$, and $p = \infty$. Then

$$T_1^r(w) = |w|, \text{ and } S_1^r(w) = 1 + w^2.$$

On the other hand

$$U^0(w) = 2 - w^2, \text{ and } V^0(w) = 2w.$$

Hence

$$r_1 = \inf_w \left[\max \left\{ \left| \frac{V^0(w)}{T_1^r(w)} \right|, \left| \frac{U^0(w)}{S_1^r(w)} \right| \right\} \right] = \inf_w \left[\max \left\{ 2, \left| \frac{2 - w^2}{1 + w^2} \right| \right\} \right] = 2, \text{ and } r_2 = 1.$$

So in this example the stability radius is determined by r_2 which is strictly less than r_1.

For the graphical robustness criteria we refer to [KJ] and [TP]. In the next two sections we consider two additional robustness problems, and briefly state main results concerning these problems.

3. l^∞ Schur stability

A polynomial $\mathbf{p}(z)$ is said to be Schur (or Schur stable) if $|z| < 1$ whenever $\mathbf{p}(z) = 0$. A set of polynomials is said to be Schur if and only if each polynomial that belongs to the set is Schur stable. Given a nominal Schur polynomial $\mathbf{p}^0(z) = (a_0^0 + jb_0^0) + (a_1^0 + jb_1^0)z + \ldots + (a_n^0 + jb_n^0)z^n$ of degree n we are seeking the maximal γ such that the family of polynomials $\mathcal{P}_\gamma$ defined by (2.1) is Schur stable. Optimization techniques provide a solution for this problem in three special important cases of $p = 1$, $p = 2$, and $p = \infty$. To state a result concerning l^∞ Schur stability some additional notations are needed. We denote

$$\max_{0\le k\le n} \frac{|V^0(\theta)\cos k\theta - U^0(\theta)\sin k\theta|}{\sum_{i=0}^{n} \alpha_i|\sin(k-i)\theta| + \beta_i|\cos(k-i)\theta|} \quad \text{by} \quad x_\infty(\theta)$$

$$\max_{n+1\le k\le 2n+1} \frac{|U^0(\theta)\cos k\theta + V^0(\theta)\sin k\theta|}{\sum_{i=0}^{n} \alpha_i|\cos(k-i)\theta| + \beta_i|\sin(k-i)\theta|} \quad \text{by} \quad y_\infty(\theta)$$

$$\max\{x_\infty(\theta),\ y_\infty(\theta)\} \quad \text{by} \quad d_\infty(\theta)$$

$$\min\left\{\inf_{0\le\theta\le 2\pi} d_\infty(\theta),\ \max\left\{\left|\frac{a_n^0}{\alpha_n}\right|,\ \left|\frac{b_n^0}{\beta_n}\right|\right\}\right\} \quad \text{by} \quad r_S(\infty, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

The robust stability conditions are given next:

Theorem 3.1 Suppose that the nominal polynomial $\mathbf{p}^0(z)$ is Schur stable.

1. If $\gamma < r_S(\infty, \boldsymbol{\alpha}, \boldsymbol{\beta})$ then the family of polynomials $\mathcal{P}_\gamma$ is Schur stable.
2. If $\gamma > r_S(\infty, \boldsymbol{\alpha}, \boldsymbol{\beta})$ then the family of polynomials $\mathcal{P}_\gamma$ fails to be Schur stable.

For the graphical robust criterion and the treatment of polynomials with real coefficients we refer to [PT].

4. l^p robust axis positivity

A polynomial $\mathbf{p}(z)$ is axis positive (see e.g., [S]) if

$$\mathbf{p}(s) > 0 \qquad \forall s \in R_+.$$

A set of real polynomials is axis positive if and only if each polynomial that belongs to the set is axis positive. Given an axis positive nominal polynomial $\mathbf{p}^0(z) = a_0^0 + a_1^0 z + \ldots + a_n^0 z^n$ of degree n, $n+1$ positive numbers α_k, $k = 0, 1, \ldots, n$, and $p \in [1, \infty]$ we are seeking maximal $\gamma \ge 0$ such that the family of polynomials $\mathcal{P}_\gamma^r$ defined by (2.20) is axis positive. The "zero exclusion criterion" combined with optimization techniques yields the following result:

Theorem 4.1 Suppose that the nominal polynomial $\mathbf{p}^0(z)$ is axis positive.

1. If
$$\gamma < \inf_{s\ge 0} \frac{\mathbf{p}^0(s)}{[\sum_k \alpha_k^q s^{kq}]^{1/q}},$$
then the family $\mathcal{P}_\gamma^r$ is axis positive.

2. If

$$\gamma > \inf_{s \geq 0} \frac{\mathbf{p}^0(s)}{[\sum_k \alpha_k^q s^{kq}]^{1/q}},$$

then the family $\mathcal{P}_\gamma^r$ fails to be axis positive.

5. Conclusions

In this paper applications of optimization techniques to robustness problems are illustrated by examples. The optimization approach to robustness analysis is not new, and references to optimization techniques can be found in almost every other paper on the subject. Nevertheless I believe that powerful tools of optimization theory can be further exploited for analyzing robustness problems, in particular when Kharitonov type extreme point methods fail.

6. References

[FD] Frazer R.A., Duncan W.J., On the criteria for stability for small motions. Proceedings of the Royal Society A, Vol. 124, pp. 642-654, 1929.

[KJ] Katbab A., Jury E.I., Generalization and comparison of two recent frequency–domain stability robustness results. Int. J. Contr., Vol. 53, No. 2, pp. 463-475, 1991.

[PT] Polyak B.T., Tsypkin Ya.Z., Robust stability of the linear discrete systems. Dokl. Acad. Nauk, Vol. 316, No. 4, pp. 842-845, 1991, (in Russian).

[QD] Qiu L., Davison E.J., A simple procedure for the exact stability robustness computation of polynomials with affine coefficient perturbations. Systems and Control Letters, Vol. 13, pp. 413-420, 1989.

[R] Rockafellar R.T., Convex Analysis. Princeton University Press, 1970.

[S] Šiljak D., Polytopes of nonnegative polynomials. Proceedings of the American Control Conference, Pittsburgh, pp. 193-199, 1989.

[TP] Tsypkin Ya.Z., Polyak B.T., Frequency domain criteria for l^p–robust stability of continuous linear systems. To appear in IEEE Trans. Autom. Contr.

Testing Robust Stability of General Matrix Polytopes is an NP-Hard Computation

Gregory E. Coxson and Christopher L. DeMarco
Department of Electrical and Computer Engineering
University of Wisconsin-Madison
1415 Johnson Dr., Madison, WI 53706
dols@cae.wisc.edu

In recent years, considerable effort has been devoted to a class of computational problems that are often termed "robust stability with structured perturbations." This literature has typically focused on system models represented by linear time invariant ordinary differential equations, and has considered families of systems that result when some underlying real parameters are allowed to take values from a specified set. Extrapolating from the case of a single fixed system model, an obvious question relating to such families of systems is whether or not every member of the family maintains root or eigenvalue locations in a specified region. For example, one might seek an efficiently computable test to determine if all roots/eigenvalues for all systems in the family have strictly negative real parts. This is the typical formulation of the robust stability problem for continuous-time systems, and will be the focus of this paper.

This paper considers families of linear systems described in state space normal form, where the set of possible system matrices is given by a polytope in $R^{n \times n}$. The goal is then to show that a general algorithm to decide if there exist matrices in the polytope having eigenvalues with non-negative real part would solve a known NP-complete problem as a special case; i.e., this robust stability test involves a computation at least as costly as that required for the class of NP-complete problems.

The NP-complete problem employed here is termed "ONE-IN-THREE 3SAT" by [1], and is a decision problem in Boolean logic. Following the style of presentation in [1], the problem may be defined by the structure of data necessary to specify an "instance" (a particular case), and the question relating to this data that must be decided. For ONE-IN-THREE 3SAT, these are:

Instance: A set of n Boolean variables U, and a collection C of k disjunctive clauses over U such that each clause $c \in C$ has $|c| = 3$ (i.e., each clause involves exactly three distinct variables selected from U).
Question: Is there a truth assignment for variables from U such that each clause in C has exactly one true literal?

A truth assignment for the Boolean variables in U has an obvious association with an n-dimensional vector u, where each element in u has a value of 0 or 1. Any 3SAT clause has the form $c = l_1(x) \vee l_2(y) \vee l_3(z)$, where $x, y, z \in U$ and for each i, l_i maps its argument symbolically either to itself or to its negation. A truth assignment satisfies clause c in the sense of ONE-IN-THREE 3SAT if and only if it satisfies

$$(l_1(x) \wedge \overline{l_2(y)} \wedge \overline{l_3(z)}) \vee (\overline{l_1(x)} \wedge l_2(y) \wedge \overline{l_3(z)}) \vee (\overline{l_1(y)} \wedge \overline{l_2(y)} \wedge l_3(z)). \qquad (1)$$

By a mapping similar to one used in [2], it is possible to derive from (1) a multilinear characteristic function associated with the set of truth assignments which satisfy the clause in the sense of ONE-IN-THREE 3SAT. The sum of the characteristic functions of the clauses in C is a function $p(u)$ with the property that it achieves a maximum of k over $[0, 1]$ if and only if the desired truth assignment exists. If a satisfying truth assignment does not exist, then $p(u) \leq k - 1$ for all $u \in [0, 1]^n$. Therefore, if there exist any points in $[0,1]^n$ such that $p(u)$ achieves a value in $(k-1, k]$, the desired truth assignment exists.

The final step of the argument constructs a matrix function $M : U \rightarrow R^{m \times m}$, $m = 4(3k+1)$, with the following properties:

i. the image of U under M is a polytope, and
ii. $\det[sI - M(u)] = (s+d)^{m-5}((s+d)^5 - p(u))$, with d an arbitrary real constant.

The characteristic polynomial has 6 distinct roots, $-d$ and five roots of the form $-d+r_i$, where r_i, for $1 = 1, \ldots, 5$ is one of the five distinct fifth roots of $p(u)$. By choosing $d = (k-1+\epsilon)^{1/5}$, $0 < \epsilon < 1$, it follows that a root exists in the closed right half plane for some $u \in [0,1]^n$ if and only if the original ONE-IN-THREE 3SAT problem has a satisfying truth assignment.

Suppose that one had available a polynomial-time algorithm for determining if an arbitrary matrix polytope had any unstable elements. Given any instance of the ONE-IN-THREE 3SAT problem one could create (with a polynomial-time complexity construction) the associated matrix polytope. Deciding whether or not this matrix polytope had any elements with unstable eigenvalues would determine whether or not there exists a satisfying truth assignment for the original instance of ONE-IN-THREE 3SAT. This establishes matrix polytope stability as an "NP-hard" problem. Unless an unexpected breakthrough shows the class of NP problems to be solvable in polynomial time, any algorithm for determining robust stability of general matrix polytopes will have worst-case computational complexity that grows faster than any polynomial in the problem size.† Given the current state of the art for the class of NP-complete problems, such a development appears unlikely.

The result outlined here does not rule out the possibility that for certain important subclasses of matrix polytopes, necessary and sufficient conditions for robust stability may be computable with polynomial-time algorithms. Moreover, as with many optimization problems that have been proven to be NP-hard, research into efficient approximate algorithms for the general case must continue. For the general matrix polytope robust stability problem, this work will probably take the form of finding efficient algorithms to test stronger *sufficient* conditions for robust stability.

Acknowledgments

The authors would like to thank Dr. Donald Coppersmith of IBM Thomas J. Watson Research Laboratories, and Professors Prason Tiwari and Anne Condon of the University of Wisconsin-Madison Department of Computer Sciences for their helpful insights into relevant results in Complexity Theory. Thanks are also due Professor B. Ross Barmish for valuable discussions regarding the state of the art in robust stability problems. The support of the National Science Foundation under grant ECS-8857019 is also gratefully acknowledged.

References

[1] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman and Company, NY, 1979.

[2] Plaisted, David A., "The Application of Multivariate Polynomials to Inference Rules and Partial Tests for Unsatisfiability", *SIAM J. Comput.*, vol. 9, no. 4, November 1980.

---

† For the robust stability problem, the key parameters defining problem size will be the dimension of the matrices in question, and the dimension of the polytope (typically the number of unknown parameters).

ON STABILITY ROBUSTNESS ANALYSIS OF SYSTEMS WITH MULTIPLE LARGE PARAMETER VARIATIONS AND MULTIPLE NONLINEARITIES

Mansour Eslami
Dept. of Electrical Engineering and Computer Science (M/C 154)
The University of Illinois at Chicago
Chicago, Illinois 60680-4348

Abstract. In this paper we present a unified method to construct "the largest" class of robustly stable matrices having multiple large parameter variations. This construction enables us to establish the stability robustness properties of linear time-invariant systems with multiple large parameter variations and a large class of nonlinear dynamic systems with both multiple large parameter variations and multiple nonlinearities.

1. INTRODUCTION

To establish the necessary and sufficient conditions of the stability robustness properties of linear time-invariant and a large class of nonlinear dynamic systems is directly related to that of a stable matrix having multiple large parameter variations. Because, in general, most dynamic systems of interest to us are modeled by a set of ordinary differential equations *(ODE's)* which are characterized in a matrix form. In this article we present a unified method to cover this general problem from theoretical and computational point of view. This methodology yields a class of robustly stable dynamic systems whose parameters (coefficient matrices) are changing in a hyperbox (we call *the largest* or *the most extended* such hyperbox the *E-Box*) in the "parameter variations space" *(PVS)*. The main thrust of this study is to construct this hyperbox or its best estimate which we call the stability robustness measure *(SRM)* of the system. Generally speaking the *SRM* is a convex set that is stacked appropriately *under* the *E-Box*. This method of construction is in contrast with a vast research in this area which is reported in the engineering literature.

First we emphasize that by robustness we mean a situation that corresponds to *large* variations of parameters where the classical sensitivity analysis that inherently deals with small parameter variations is no longer valid. The prerequisite to any meaningful robustness study of a dynamic system is to indicate a particular and a desirable system behavior, and to specify the precise perturbation mechanism that may not disturb this property. In this paper our analysis is pertained to the system stability and we want to know how we can perturb a system *around a given stable operating mode* without deteriorating its stability. The state space representation of the feedback system that is considered in this paper is described by a standard set of *ODE's* which this set can also be considered as the linearized version of a set of nonlinear differential equations.

$$\dot{x} = (A + BK)x \triangleq S(\theta)x. \qquad (1)$$

Here θ is a generic name for the vector of system *parameters* and we note that A, B, and K are assumed to be matrices. The main implication of this representation is that the construction of the largest class of systems will encompass the variations in system parameters A and B as well as the design or the linearization parameters K. When (1) is the linearized version of the model of a nonlinear system, then K being a matrix means that the system has multiple nonlinearities [5]-[7]. In this study we present the essential steps toward robustness analyses of such systems.

We may formulate the stability requirement of (1) in matrix form (Lyapunov approach) or in polynomial form (characteristic equation) at any given operating condition θ^o. As is shown in [2], however, upon selecting θ^o and the pertinent $\Delta\theta$, then there are a number of possibilities regarding the structure of ΔS, which can be categorized into two main groups of single- and multiple-parameter variations. These structured matrices are subsequently become linear or nonlinear functions of $\Delta\theta$.

2. STABILITY ROBUSTNESS ANALYSIS

Our general analysis stems from the following theorem which provides a set of sufficient conditions for asymptotic stability of a perturbed system.

Theorem 1 [4]: Consider $\dot{x}(t) = Sx(t)$, where $S \in R^{n\times n}$ is an asymptotically stable matrix satisfying the Lyapunov equation $S^TP + PS + Q = 0$, with $Q = Q^T > 0$ and $P = P^T > 0$. Let $S \rightarrow S + \Delta S$, then $S + \Delta S$ remains asymptotically stable if

$$\Delta S\, Q^{-1}\, \Delta S^T < \tfrac{1}{4} P^{-1}\, Q\, P^{-1}, \text{ or} \tag{2}$$

$$\Delta S^T\, P\, Q^{-1}\, P\, \Delta S < \tfrac{1}{4} Q. \tag{3}$$

If the Lyapunov equation is set as $SP + PS^T + Q = 0$, then the above two inequalities become, respectively, as follows.

$$\Delta S^T\, Q^{-1}\, \Delta S < \tfrac{1}{4} P^{-1}\, Q\, P^{-1}, \tag{4}$$

$$\Delta S\, P\, Q^{-1}\, P\, \Delta S^T < \tfrac{1}{4} Q. \tag{5}$$

Comment 1: Clearly the above two Lyapunov equations yield different solutions for P and different "solutions" for ΔS. Also in a given problem we do not know *a priori* which of the inequalities (2) to (5) yield the "best" ΔS as shown by numerical examples in [2]-[3]. Note that by $N \geq M$, for $0 < N = N^T \in R^{n\times n}$ and $0 \leq M = M^T \in R^{n\times n}$, we mean $N - M \geq 0$.

Although Theorem 1 applies to all sorts of perturbational matrices ΔS's, corresponding to a stable S, we note that depending on the structure of ΔS, we need to classify the problem into two major groups, namely, the single- and the multiple-parameter variations.

2.1. Single-Parameter Variations

In order to construct ΔS from any one of (2) to (5), when in general we have $M < N$ for two symmetric matrices $M \geq 0$ (which is unknown) and $N > 0$, we need to specify a particular norm. To solve for $M < N$ we propose to set the $\lambda_{max}(M)$ ($\triangleq$ the maximum eigenvalue of M) $\leq \lambda_{min}(N)$ ($\triangleq$ the minimum eigenvalue of N). A more conservative choice than this one is to set $Tr(M)$ ($\triangleq$ sum of all eigenvalues of M) $< \lambda_{min}(N)$. If the above N and M are not diagonal, then we can use the simultaneous diagonalization theorem to diagonalize these matrices [4]. In this case $\Delta S = \varepsilon(\Delta\theta)\Delta S'$, where $\varepsilon(\Delta\theta)$ is in effect a single parameter and $\Delta S'$ is a known matrix.

Comment 2: The upper bounds (2) to (5) are only sufficient conditions for an asymptotically stable matrix S and these upper bounds are functions of $Q = Q^T > 0$. In both single- and multiple-parameter variations there are two major issues regarding the applications of these upper bounds.

(i) Depending on a given problem, for example, in a single-parameter variations we have, $\Delta S \triangleq \varepsilon(\Delta\theta_1)\Delta S'$, where $\varepsilon(\Delta\theta_1)$ is in effect one single-parameter that is a function of the physical parameter of the system, and $\Delta S'$ is a numerical matrix which is constructed at a given operating point based on the information we may have regarding the parameter variations scheme. In these cases the construction of *the largest* ΔS can be achieved unambiguously since we only have one scalar parameter $\varepsilon(\Delta\theta_1)$. For those cases in which we have a set of varying parameters, and/or ΔS can not be uniquely defined (for example, a case in which one single-parameter appears nonlinearly in ΔS), then we use the term *net* and we search for a net ΔS. This is generally done by letting $\Delta S = \varepsilon\Delta S'$, where $\Delta S'$ is a *candidate* for ΔS, and we search for the best ε in order that $\varepsilon\Delta S'$ becomes an *admissible* ΔS. From the computational point of view, we always let $\Delta S \triangleq \varepsilon\Delta S'$ for a known $\Delta S'$ and search for the *largest* ε.

(ii) Furthermore since the upper bounds (2) to (5) are functions of $Q = Q^T > 0$. Therefore we need to introduce a method that generates *the largest* upper bound of ΔS independent of the initial

choice of $Q = Q^T > 0$. Since the actual stability robustness measure of a dynamic system is, of course, independent of any computational or physical measurement methods that are used to generate that, therefore this dependency should somehow be eliminated. To that end we propose to choose any arbitrary initial $Q = Q^T > 0$ in order to perturb the system. If the system remains stable, then we use the same Q and we perturb it again repeatedly. This iterative method, as formulated in an algorithm in [4] is an alternative approach to respond to this dependency problem. As a consequence of this algorithm we let $Q = I$ (or 2I), to avoid extra matrix manipulations. Note that both I or 2I results in the same upper bound in (2) to (5) for each step of perturbation.

2.2. Multiple-Parameter Variations

We now give some additional insights into the construction of robust $S + \Delta S$ for systems with multiple varying parameters. But first we recall that the constructional procedure that is suggested in Comment 2, when applied to these cases will result in *an* admissible ΔS which may be *an* acceptable perturbation matrix in *a* given problem. However, since this construction is based on one candidate (i.e., one $\Delta S'$) or one set of net parameter variations, therefore it may not always be the best answer and it may not be clear how that perturbation scheme will be accomplished when all of the varying parameters are changing. For example, when we have a point in the *PVS*, then there are infinitely many possible paths for parameter variations that will reach from the origin, *per se*, to that point. Certainly the system does not remain stable for all the points on all of these paths. On the other hand, if we choose another candidate for ΔS (i.e., a new $\Delta S'$) and we search for an admissible ΔS (i.e., a new ε) we will end up having a completely different ΔS. In the following we show that there exists a connected set of allowable parameter variations whose corresponding ΔS's will yield a class of asymptotically stable systems when all of the varying parameters are changing in that set. In this frame of work we will attribute the word "largest" to the above set, i.e., a set of points in the *PVS*, whose corresponding ΔS's yield a class of $S + \Delta S$ which is asymptotically stable. When this construction is completed and when we have found the "largest" set, then naturally all of the other results that are reported in the literature for specific matrix norms will be included in this set as well. Again we note well that the crucial issue is to expedite the construction of such set of points, we really are not improving the *E-Box* of the system. This set is invariant for any given system and we are just constructing that set as best as we can.

To search for the largest set of parameter variations we note that since there are infinitely many possible points in such set of parameter variations *(around a given stable operating point)*, therefore it is essential to have a procedure to construct this set or the corresponding class of robust systems from the information which is stemming from a finite number of admissible ΔS's. That construction is indeed possible as the result of next theorem which establishes the nucleus of our methodology.

Theorem 2 (E-Box)[3]: Consider (2) with $Q = 2I$ that is repeated below.

$$\Delta S(\Delta\theta)\, \Delta S^T(\Delta\theta) < P^{-2}(\theta). \qquad (6)$$

Suppose from this inequality we have extracted, in principle, the 2^r "solutions": $\Delta S_1, \Delta S_2, \ldots, \Delta S_{2^r}$, each corresponding to one point (or vertex) in the *PVS*, which form a hyperbox in that space. If for each and every points in this hyperbox we can express the corresponding ΔS in terms of the above ΔS_i's as follows.

$$\Delta S = \sum_{i=1}^{2^r} w_i \Delta S_i, \quad \text{with} \quad \sum_{i=1}^{2^r} w_i = 1, \quad 0 \le w_i < 1, \ \text{for all i.} \qquad (7)$$

Then each and every points of this convex hull corresponds to an asymptotically stable $S + \Delta S$. (We call *the largest or the most extended* such hyperbox the *E-Box*.)

Remark 1: Clearly (7) is true for all linear perturbation matrices including the polynomials. For nonlinear perturbation matrices, we need to study each problem individually. In this situation we can approximate the nonlinear perturbation matrix, using a Taylor series expansion, with a set of linear perturbation matrices that is computed for *small* parameter variations in order to comply with (7). Thus Theorem 2 can be used for those connected nonlinear perturbation matrices that can be expanded by a Taylor series at each operating point.

Remark 2: As a result of Theorem 2 we can construct a hyperbox in the *PVS* whose corresponding ΔS's result in $S + \Delta S$ that is asymptotically stable. Then we can show that using an iterative approach similar to that of single-parameter variation we can extend the previous hyperbox in order to reach to the *largest* such hyperbox which we call the *SRM*.

Example 1: Consider the following asymptotically stable matrix

$$S(\equiv S_1) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & -1 & -4 \end{bmatrix}, \tag{8}$$

whose eigenvalues are $\lambda_{1,2} = \tfrac{1}{2}(-1 \pm j\sqrt{3})$ and $\lambda_{3,4} = -2 \pm \sqrt{3}$. First we assume that this system is perturbed according to

$$\Delta S = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \varepsilon & -\varepsilon & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -\varepsilon & \varepsilon & 0 \end{bmatrix}. \tag{9}$$

This example (due to Professor D.L. Russell) comes from a fourth-order differential equation characterizing a mechanical model of an accelerometer. From the physics of this system we know *a priori* that as $\varepsilon \to 1$, then the system becomes unstable. Therefore we want to compute the *largest* ε such that $S + \Delta S$ remains asymptotically stable. We have applied the iterative method of [4] (with $Q = I$) to this example, and after five iterations we have $\varepsilon = 0.797481 < 0.8$ (0.8 is the critical value of ε found using Routh-Hurwitz criterion) and the system remains asymptotically stable.

Now suppose that the perturbation policy is according to the following scheme.

$$\Delta S = \Delta\theta_1 \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} + \Delta\theta_2 \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix}. \tag{10}$$

In other words, instead of having one parameter variation ε we have two parameter variations $\Delta\theta_1$ and $\Delta\theta_2$ that are acting independently on ΔS. Our goal is to apply our method to this case and to construct an admissible set of parameter variations such that for this set the corresponding class of dynamic systems remains stable (or asymptotically stable).

Solution: For the sake of illustration first we assume that $\Delta\theta_j > 0$, for all j. Then we search for the largest variations along all the axes and the "45^o line". In this example, using single-parameter variation algorithm we can show that

$$O_1 \triangleq (\Delta\theta_1 = \Delta\theta_2 = 0), \qquad A \triangleq (\Delta\theta_1 = 0, \Delta\theta_2 = 2.06),$$

$$C_n^{++} \triangleq (\Delta\theta_1 = \Delta\theta_2 = 0.8), \qquad B \triangleq (\Delta\theta_1 = 1.0, \Delta\theta_2 = 0), \tag{11}$$

are stable (extreme) points. As we continue to study this problem, we realize that perhaps the only question that we need to answer in this example (which has two varying parameters) is the following. Namely, on which paths between C_n^{++} to A or C_n^{++} to B the system remains stable? The following analysis which is based on Theorems 1 & 2 is intended to answer this question. Let $J = 1, 2, \ldots, N$, be the iteration number.

STEP 0 (To update S*):* We let $S_J = S_{J-1} + \Delta S_{J-1}$ with $S_1 \equiv S_0 \equiv S$ being the initial asymptotically stable matrix and $\Delta S_0 \equiv 0$.

STEP 1 (To solve for the Lyapunov equation): For the current S_J we solve for P_J according to $S_J^T P_J + P_J S_J + 2I = 0$. (*At* $J = 1$, and in a typical application we may also solve for the other Lyapunov equation in Theorem 1 in order to study which perturbation scheme of (2) to (5) should be utilized in a given problem. In that case the following steps must be changed accordingly.) At $J = 1$ we have

$$P_1 = \frac{1}{10}\begin{bmatrix} 31 & 11 & 2 & 1 \\ 11 & 24 & 9 & 3 \\ 2 & 9 & 45 & 10 \\ 1 & 3 & 10 & 5 \end{bmatrix}. \tag{12}$$

STEP 2 (To construct $\Delta S \Delta S^T$): Given any structured ΔS we can easily compute $\Delta S \Delta S^T$ which will symbolically remain the same throughout the analyses. Here

$$\Delta S \Delta S^T = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \Delta\theta_1^2 + \Delta\theta_2^2 & 0 & -2\Delta\theta_1\Delta\theta_2 \\ 0 & 0 & 0 & 0 \\ 0 & -2\Delta\theta_1\Delta\theta_2 & 0 & \Delta\theta_1^2 + \Delta\theta_2^2 \end{bmatrix}. \tag{13}$$

STEP 3 (To interpret the sufficient condition): Using the sufficient condition $\Delta S \Delta S^T < P_J^{-2}$ we need to establish a set of inequalities which yields sufficient conditions for our parameter variations. Here we look for conditions on $\Delta\theta_1$ and $\Delta\theta_2$ such that $(P_J^{-2} - \Delta S \Delta S^T) > 0$ holds. At $J = 1$ we have

$$0.18423 > 0, \tag{14a}$$

$$\Delta\theta_1^2 + \Delta\theta_2^2 < 0.196, \tag{14b}$$

$$\Delta\theta_1^2 + \Delta\theta_2^2 < 0.188, \tag{14c}$$

$$(\Delta\theta_1^2 - \Delta\theta_2^2)^2 + 0.117 - 1.33(\Delta\theta_1^2 + \Delta\theta_2^2) + 1.316\Delta\theta_1\Delta\theta_2 > 0. \tag{14d}$$

STEP 4 (To search for vertices): The final set of the admissible parameter variations must satisfy the intersection of all these inequalities, i.e., in this example we must have:

$$\Delta\theta_1^2 + \Delta\theta_2^2 < 0.188, \tag{15a}$$

$$(\Delta\theta_1^2 - \Delta\theta_2^2)^2 + 0.117 - 1.33(\Delta\theta_1^2 + \Delta\theta_2^2) + 1.316\Delta\theta_1\Delta\theta_2 > 0. \tag{15b}$$

From these inequalities we must extract $2^r = 2^2 = 4$ sets of points which satisfy (15). Obviously and from (15a), we must search for the admissible parameter variations which are inside of the circle of radius 0.43. Certainly the following points which are directly computed from (15) are possible candidates for the vertices of the desired set:

$$O_1 \triangleq (\Delta\theta_1 = \Delta\theta_2 = 0), \qquad A_1 \triangleq (\Delta\theta_1 = 0, \Delta\theta_2 = 0.295),$$

$$C_1 \triangleq (\Delta\theta_1 = \Delta\theta_2 = 0.295), \quad B_1 \triangleq (\Delta\theta_1 = 0.295, \Delta\theta_2 = 0). \tag{16}$$

Substituting any of these values for parameter variations in (10) will yield the corresponding ΔS's.

STEP 5 (To construct an admissible set): Connecting the above four vertices of (16) and because of Theorem 2 we will have a set of admissible points for parameter variations which is shown in Fig. 1. At this juncture it is imperative to recall that the above four vertices are generated from the *same* P_1, and all of the requirements of Theorem 2 are satisfied, therefore we are able to connect these four vertices and to construct this set.

Comment 3: Clearly and as we expect the set of admissible points that is generated from the analytical procedures in Steps 3 and 4 is not the largest possible set. Because from the first part of this example, we have already established the fact that the point, say, $C_n^{++} \triangleq (\Delta\theta_1 = \Delta\theta_2 = 0.8)$ results in a stable perturbed system. Therefore we now search for the ways of expanding our initial set of admissible points in the *PVS* and that is the purpose of the next step.

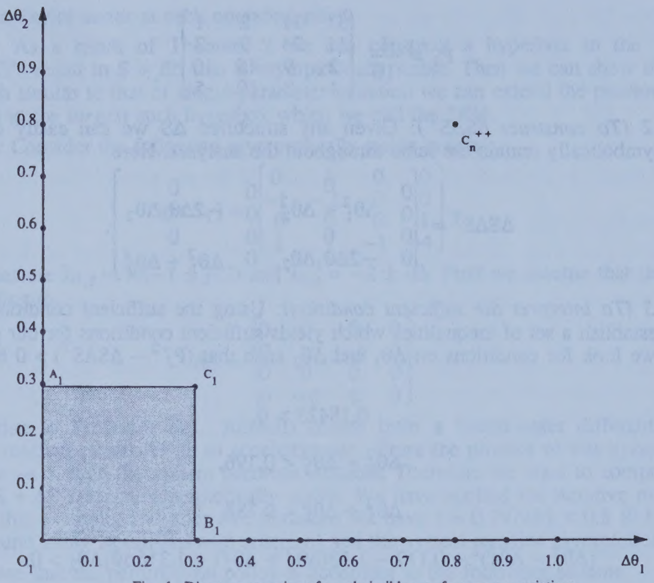


Fig. 1. Direct construction of an admissible set of parameter variations.

Iteration: J+1

STEP 6 (The construction of the most extended set): As in the case of the single-parameter variation we now propose to translate the coordinates of the *PVS* to the farest vertex of the convex hull in order to construct more regions of the admissible points in that space. That means we now start with $S_{J+1} = S_J + \Delta S_J$ (Step 0), where the corresponding ΔS_J is computed in Step 4, then we go back to Step 1. Certainly, some of the parameter variations may have negative signs in a given set of coordinate in the *PVS* and we must be alert of this fact, which is equivalent of reducing the value of one parameter variation while changing the other one. We must consider this situation very carefully since its implication is that, as we progress to translate the coordinates in different directions we must search for points in the preceding set in order to connect the two consecutive convex sets *as smoothly as* we possibly can, and to cover the entire allowable region of admissible points in the *PVS* as smoothly and completely as we can. We summarize our computations in the following and as we progress we draw each corresponding subsets of admissible parameter variations in Fig. 2a.

At J = 2: From the result in Step 4 we translate the coordinates to point C_1. Therefore now $S_2 = S_1 + \Delta S_1$, where $\Delta S_1 = \Delta S(\Delta\theta_1 = \Delta\theta_2 = 0.295)$. For this S_2 and from Step 2 we have the following sufficient conditions for $(P_2^{-2} - \Delta S \Delta S^T) > 0$

$$\Delta\theta_1^2 + \Delta\theta_2^2 < 0.112, \tag{17a}$$

$$(\Delta\theta_1^2 - \Delta\theta_2^2)^2 + 0.141 - 0.927(\Delta\theta_1^2 + \Delta\theta_2^2) + 0.303\Delta\theta_1\Delta\theta_2 > 0. \tag{17b}$$

We again let $\Delta\theta_1 = \Delta\theta_2$. Then we can show that for values such that $|\Delta\theta_1| = |\Delta\theta_2| \le 0.236$ these inequalities hold. Therefore we select the following four new vertices which are shown in Fig. 2a:

$$C_2^{++} \triangleq (\Delta\theta_1 = \Delta\theta_2 = 0.236), \quad C_2^{-+} \triangleq (\Delta\theta_1 = -0.236,\ \Delta\theta_2 = 0.236),$$

$$C_2^{--} \triangleq (\Delta\theta_1 = \Delta\theta_2 = -0.236), \quad C_2^{+-} \triangleq (\Delta\theta_1 = 0.236, \Delta\theta_2 = -0.236). \tag{18}$$

At J = 3: Continuing in this fashion and from the above results we now translate the coordinates of the *PVS* to point C_2^{++}. Therefore now $S_3 = S_2 + \Delta S_2$, where $\Delta S_2 = \Delta S(\Delta\theta_1 = \Delta\theta_2 = 0.236)$, and from Step 2 we have P_3. The sufficient condition for the asymptotic stability of S + ΔS yields the following four new vertices:

$$C_3^{++} \triangleq (\Delta\theta_1 = \Delta\theta_2 = 0.113), \quad C_3^{-+} \triangleq (\Delta\theta_1 = -0.113, \Delta\theta_2 = 0.113),$$

$$C_3^{--} \triangleq (\Delta\theta_1 = \Delta\theta_2 = -0.113), \quad C_3^{+-} \triangleq (\Delta\theta_1 = 0.113, \Delta\theta_2 = -0.113). \tag{19}$$

We note that the constructional method is progressing along the line $\Delta\theta_1 = \Delta\theta_2$ rather quickly. However, we want to know how to fill points on either sides of the previous line. Because, technically speaking, we can not [yet] connect C_3^{-+} to C_2^{-+} *per se.*

At J = 4: Thus we now translate backward the coordinates from O_3 (at C_2^{++}) to O_4 (at C_2^{-+}). That means $S_4 = S_3 + \Delta S_3$, where $\Delta S_3 = \Delta S(\Delta\theta_1 = -2\times0.236, \Delta\theta_2 = 0.0)$. For this S_4 and from Step 2 we have P_4 and if we keep our parameter variations in the positive quadrant of the *PVS*, then we have

$$C_4^{++} \triangleq (\Delta\theta_1 = 0.059, \Delta\theta_2 = 0.07), \quad C_4^{-+} \triangleq (\Delta\theta_1 = -0.059, \Delta\theta_2 = 0.07),$$

$$C_4^{--} \triangleq (\Delta\theta_1 = -0.059, \Delta\theta_2 = -0.07), \quad C_4^{+-} \triangleq (\Delta\theta_1 = 0.059, \Delta\theta_2 = -0.07). \tag{20}$$

Remark 3: Now the message is clear. By this iterative approach we can construct the entire set of admissible parameter variations such that for each point in this set the corresponding ΔS yields an asymptotically stable S + ΔS. Thus we have a class of asymptotically stable systems surrounding the initial system and indeed that is the true meaning of the stability robustness. This set, which can be constructed in order to become *the largest* or *the most extended* set of admissible parameter variations, is shown in Fig. 2b. Clearly, it seems that by increasing the iteration number and in a few different directions we can smooth the boundary of the previous set (Fig. 2b). However there are a few short cuts that we may undertake. For example, the perturbed system corresponding to the nonshaded set between the two sets O_4 and O_1 shown in Fig. 2b, will result in an asymptotically stable perturbed system, because of the convexity of the perturbation matrices and the fact that this set is surrounded by four other convex stable sets (including the vertical axis). The same can be said for the other boundary sets in that figure. Indeed we may conclude that if we continue the construction along the "45° line" $\Delta\theta_1 = \Delta\theta_2$ $(= \cdots = \Delta\theta_r)$ in order to reach to the point C_n^{++} (here at $\Delta\theta_1 = \Delta\theta_2 = 0.8$), then the entire square box whose farest vertex is C_n^{++} will yield an asymptotically stable matrix provided that the single-parameter variation along each axes is outside of this box. Because in this square the perturbation matrix remains a convex matrix and that is why it is important to study the type of a perturbation matrix before attempting to construct the *E-Box.* Clearly, if the extreme single-parameter variation of each axes is inside of the above square box, then we need to compress that box accordingly. However, as mentioned earlier in the introduction of solution to Example 1, the real issue is how to connect C_n^{++} to A or B of (11). Since to the best of our knowledge, there is no theorem which supports the stability of the perturbed matrix when we connect by a straight line (in order to maintain the convexity) C_n^{++} to A or B. Therefore we must appeal to this constructional method in order to expedite generating the actual *SRM* of the system. In other words, the stability of the perturbed matrix (system) may break down on this line. However, if we can find the appropriate stable paths between C_n^{++} to A or B, then we can close an area in the *PVS* (considering the axes and the "45° line") which yields an asymptotically stable S + ΔS. From the construction of this closed path and the convexity of the perturbed system inside of this closed path we can immediately conclude that for all the points

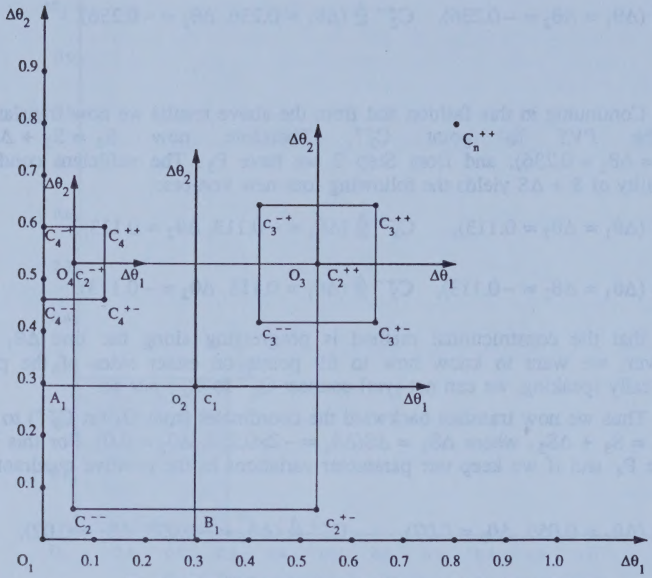

Δθ2
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
O1
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Δθ1
Cn++
Δθ2
Δθ2
Δθ2
C4-+
C4++
O4
C2-+
Δθ1
C4--
C4+-
C3-+
C3++
O3
C2++
Δθ1
C3--
C3+-
A1
O2
C1
Δθ1
C2--
B1
C2+-

Fig. 2a. Toward the construction of the *largest* set of parameter variations.

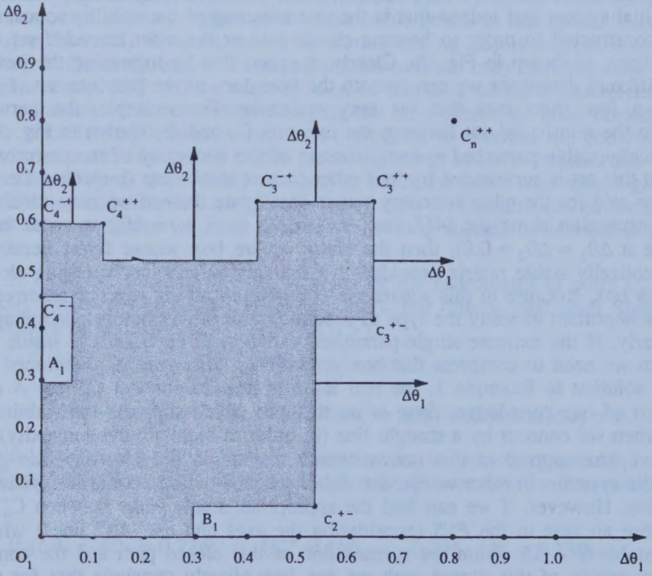

Δθ2
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
O1
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Δθ1
Cn++
Δθ2
Δθ2
C4-+
C4++
C3-+
C3++
Δθ1
C4--
C3+-
A1
Δθ1
B1
C2+-

Fig. 2b. Toward the construction of the *largest* set of parameter variations.

inside of that set the perturbed system remains asymptotically stable. From all of these thoughts we conclude that in general when we have unidirectionally found the extreme points A, B, C_n^{++}, *etc.*, then our computational efforts must be concentrated on finding stable paths (or sets) connecting these extreme points.

Remark 4: There are a few other observations which we must make in order to apply this method to system with more than two varying parameters [3].

First of all, the above detailed numerical example shows that as we reach to point C_n^{++} the "area" of each corresponding stable set shrinks. We also note that the "radius" of each of these sets is proportional to $\lambda_{min}(P_J^{-2})$, where J is the iteration number. Indeed to study the propagation of the solutions to the corresponding Lyapunov equations in this constructional method is a very informative problem. Since also the length of $O_1C_n^{++}$ is finite (otherwise we need infinite iterations which in that case we must completely modify this procedure), and this can be computed from the classical Routh-Hurwitz method. Therefore an upper bound for the iteration number along this line is the ratio of $O_1C_n^{++}/\lambda_{min}(P_n^{-2})$. Of course, here we note that we should allow a reasonable numerical tolerance for reaching to C_n^{++}. Since we may never computationally reach to the theoretical C_n^{++} (in this example we stop at $0.797481 < 0.8$), and therefore P_n will also be corresponding to this practical value. Otherwise at the very limit, the $\lambda_{min}(P_n^{-2})$ will approach zero, i.e., the number of iterations will become infinite. Clearly we should avoid this situation which can occur in a typical iterative algorithm. Similar estimates for iteration numbers along the boundary lines can be made. All in all the total number of iterations for a *typical* application is finite if we maintain a reasonable numerical tolerance.

Secondly, we recall that we open the solution to Example 1 with an assumption that $\Delta\theta_j > 0$ for all j. Naturally, when we complete our search in the first quadrant of the *PVS*, then we should repeat this search in the next quadrant of that space.

Thirdly, we recall that in our example we had only two parameters. In general we have r parameters and we must find all of the extreme points first. Then we must look for ways of connecting these points such that on each path (set) the system remains stable. We note that in general when $r > 2$, then the "45° line" must be further studied. Here when we reach to the C_n^{++}, then we must search for unidirectional extreme points in the sub-spaces of the *PVS*.

Comment 4: To construct inequalities such as (15a) and (15b) in a typical application is a nontrivial task involving cumbersome algebra. What we propose as an alternative method for these situations is that suggested in Comment 2 and is repeated below. We initially make a numerical guess for parameter variations $\Delta\theta_1$ and $\Delta\theta_2$, *etc.*, and we substitute these values in ΔS. This substitution will make the parametric ΔS to become a numerically defined perturbation matrix which we call that $\Delta S'$ and it is a *candidate* for ΔS. Then we let $\Delta S = \varepsilon\Delta S'$ and look for the best or the largest ε. When this ε is found then all of these guesses will be scaled accordingly and we get one point in the *PVS* whose corresponding perturbed system is stable.

2.3. Multiple Nonlinearities

Consider a class of nonlinear dynamic systems described by

$$\dot{x}(t) = Ax(t) + Bf(\sigma(t)), \quad \sigma(t) = c^T x(t), \tag{21}$$

where $x(t) \in R^n(t)$ is the state vector, $f(\sigma(t)) \in R^m(t)$ is a vector of system nonlinearities, A and B are system coefficient matrices. This particular model represents a large class of nonlinear dynamic systems which is of interest to engineers. This nonlinear vector also represents inaccuracies in modeling of a dynamic system. There is a wealth of literature on this problem [3]. References [1]-[3], [5]-[10], are just representative. Here when $f(\sigma(t))$ is replaced by a linear term $Kx(t)$, $K \in R^{m \times n}$, then (21) becomes the same as (1) which is a linearized version of (21). In [2] an in-depth analysis of nonlinear systems with *one* nonlinearity is reported. The main issue that has been reported in [2] is that, given an operating point, how to find *the largest* class of asymptotically stable linear systems which surrounds the nonlinear system with one nonlinearity. If we can not confirm that the nonlinear system is globally 'boxed' in a sector. Then we can not definitely conclude that the linearization of the nonlinear system yields information on global stability of the nonlinear system [10]. These issues are discussed in [2] and from that analysis we can develop the largest sector which approximates the nonlinear system (with one nonlinearity), from the stability point of view, with a linear system.

The same procedure that is used in [2] for single nonlinearity case can be utilized for a system with multiple nonlinearities in order to construct *the largest class* of asymptotically stable linear systems which surrounds the original nonlinear system. The only difference is on how to interpret the meaning of the *largest,* which is now defined according to the context of multiple-parameter perturbations that we have developed in § 2.2.

3. CONCLUSIONS

In this paper we have demonstrated the generality and the breadth of a method introduced in [4], which constructs the largest class of robustly stable dynamic systems that surrounds the initial stable dynamic system. This method that is based on the Lyapunov theorem can be easily extended to several other problems of interest in control theory such as: the study of the behavior of a quadratic performance index when it is subject to a number of large parameter variations; and to certain estimation problems or covariance analysis, among other applications. This method can also be extended to Hurwitz polynomials with large coefficient (parameter) variations [3]-[4]. We are particularly interested in this extension because the rational polynomials of a complex variable with real-valued coefficients play an extremely important role in stability behavior of a set of *ODE's* and for that matter many control design problems.

4. ACKNOWLEDGMENT

The author greatly acknowledges the constructive conversation with his colleague Dr. D.S. Bayard of the JPL.

5. REFERENCES

[1] M.A. Aizerman, "Some unsolved problems in theory of automatic control and fuzzy proofs," *IEEE Trans. Auto. Contr.*, vol. AC-22, pp. 116-118, Feb. 1977.

[2] M. Eslami, "Robust nonlinear dynamic systems - The old question with a new answer," *Systems and Control Letters,* vol. 14, pp. 445-451, 1990.

[3] ________, *An Introduction to the Theory of Sensitivity in Dynamic Systems.* New York: Springer Verlag, 1992, in press.

[4] M. Eslami and D.L. Russell, "On stability with large parameter variations: Stemming from the direct method of Lyapunov," *IEEE Trans. Auto. Contr.*, vol. AC-25, pp. 1231-1234, Dec. 1980.

[5] A.I. Gaiduk, "Absolute stability of control systems with several nonlinearities," *Auto. Remote Contr.*, vol. 37, no. 6, pp. 815-821, Nov. 1976.

[6] K.S. Narendra and J.H. Taylor, *Frequency Domain Criteria for Absolute Stability.* New York: Academic, 1973.

[7] E.S. Pyatnytsky, "New research on absolute stability of automatic control systems," (Review), *Auto. Remote Contr.*, vol. 29, no. 6, pp. 855-881, June 1968.

[8] D.D. Siljak, "Parameter space methods for robust control design: A guided tour," *IEEE Trans. Auto. Contr.*, vol. AC-34, pp. 674-688, July 1989.

[9] M. Vidyasagar, "New directions of research in nonlinear system theory," *Proc. IEEE,* vol. 74, no. 8, pp. 1060-1091, Aug. 1986.

[10] J.C. Willems, *The Analysis of Feedback Systems.* Cambridge, MA: MIT Press, 1970.

ASYMPTOTIC PROPERTIES IN ADAPTIVE FILTERING

G. Yin*
Department of Mathematics
Wayne State University, Detroit, MI 48202

L. Zhao†
Department of Mathematics
SUNY/ College at Old Westbury, Old Westbury, NY 11542

Abstract. Asymptotic properties of adaptive filtering algorithms are developed in this work. First, by using elementary inequalities, the with probability 1 (w.p.1) convergence is established via a direct approach. This extends some existing results. Next, the problem of asymptotic distribution is discussed by virtue of a suitable scaling. In addition, confidence region estimates are constructed and a stopping rule is proposed and analyzed. Furthermore, optimal design issues are discussed.

Keywords. adaptive filter, convergence, asymptotic normality, stopping rule, optimal design.

1. Introduction

This work deals with a class of recursive algorithms of adaptive filtering type. We survey, update and extend some of the recent results. Let $X_n, Y_n \in \mathbb{R}^r$, $\psi_n \in \mathbb{R}$. We begin with the following algorithm:

$$X_{n+1} = X_n - a_n Y_n (Y_n' X_n - \psi_n), \tag{1.1}$$

where $\{a_n\}$ is known as gain or step size.

The above algorithm was devised to solve certain identification or optimization problems. For example, let $\{\psi_n\}$ represent a sequence of real-valued desired output (training or reference signals for instance). Let $\{u_n\}$ be a sequence of input signals and $Y_n = (u_n, \ldots, u_{n-r+1})'$. The observed output at n is given by $\bar{y}_n = \sum_{i=0}^{r-1} X_{n,i} u_{n-i}$, where $X_{n,i}$ denotes the i-th component of X_n. The problem is to adjust the parameter X_n such that the output $\{\bar{y}_n\}$ matches the signal $\{\psi_n\}$ as well as possible. A common practice is to interpret the 'optimal matching' problem in the mean square sense. For the aforementioned purposes, the most widely used algorithms are of the form (1.1). It should be pointed out that one of the crucial features in this kind of algorithms is the statistics of $\{\psi_n, Y_n\}$ are not known.

For simplicity, we shall only consider algorithm (1.1) in this paper. For a constrained version of the algorithm, we refer to the work [3, 20, 26] among others. Our main interest is to obtain asymptotic properties of the algorithms. Due to the page limitation, we are unable to spell out all the detailed proofs. Nevertheless, appropriate references or idea of proofs will be provided.

---

* Research of this author was supported in part by the National Science Foundation under Grant DMS-9022139.

† Research of this author was supported in part by SUNY College at Old Westbury under the Summer Research Grant.

The rest of the paper will be organized as follows. Some recent literature will be reviewed in the next section first. Section 3 is concerned with the convergence problems. The rate of convergence and asymptotic normality is studied in Section 4. In Section 5, a useful stopping criterion is given. The optimal design issues are discussed in Section 6. Finally, some further remarks are made in Section 7. In the sequel, K will denote a generic positive constant. Its value may change from usage to usage. z' will denote the transpose of a matrix z with appropriate dimensions.

2. Adaptive filtering–an overview

In 1960, Widrow and Hoff [25] devised the LMS (Least Mean Squares) algorithm for adaptive filters. In fact, many problems in stochastic programing and optimization can be cast into the form of (1.1), such as adaptive antenna, adaptive control, adaptive equalization, adaptive tracking, adaptive beam forming, learning and pattern recognition, recursive identification, under water signal processing and many other related problems.

The work [25] has stimulated widespread reactions for the past three decades. Adaptive filtering has become a well developed subject with a vast literature. Statistically independent signals were studied in [3, 25]. Assuming that X_n return to a compact region infinitely often, correlated signals were treated in [12, 15] via an approach known as the ODE (ordinary differential equation) method. A projected version of (1.1) was proposed in [10]. Stationary M-dependent signals with finite moments and finite memory were dealt with in [6, 17]; both with probability 1 and mean square convergence results were obtained. Adaptive beam forming with non-stationary and correlated signals were treated in [4, 25], and the almost sure convergence was established. Rate of convergence questions for adaptive filtering type of algorithms were addressed in [9, 13, 14, 20]. The methods of weak convergence were utilized to investigate the local limit behavior; asymptotic results were established by means of martingale problem formulations (cf. [13, 14, 20]). A strong invariance principle was obtained in [9]. The optimal design problem was addressed by using asymptotic covariance as a performance index in [1]. We should also point out that many important contributions have been made to the related least squares and gradient identification problems in recent years.

3. Convergence

We shall derive the convergence of the adaptive filtering algorithm in this section. We assume the following conditions hold.

(A1) Suppose that R is a symmetric positive definite matrix, $q > 0$ and Q is a vector. R, Q, q are constant but possibly unknown. $EY_nY_n' = R$, $EY_n\psi_n = Q$, $E\psi_n^2 = q$.

(A2) $\sum_{j=1}^{\infty} a_j(Y_jY_j' - R)$ converges w.p.1 and $\sum_{j=1}^{\infty} a_j(Y_j\psi_j - Q)$ converges w.p.1. The gain $\{a_n\}$ satisfies $a_n > 0$, $a_n \xrightarrow{n} 0$, $\sum_{j=1}^{\infty} a_j = \infty$.

The condition (A2) which seems to be first isolated and employed in [12] is the main averaging assumption. By means of Gauss-Markov estimation, it can be shown that the minimizer x^* is given by $x^* = R^{-1}Q$. Due to the unavailability of R and Q, the above expression for x^* is actually not as informative as it may be seen. Nevertheless, the following theorem gives a much more useful alternative.

Theorem 3.1. *Under assumptions (A1) and (A2), the sequence* $\{X_n\}$ *satisfying (1.1) converges with probability one to* x^*.

Remark: In [12], a convergence theorem was obtained under the assumption that the iterates are bounded w.p.1. A projection algorithm was designed in [10] with assumption of some *a priori* information on where x^* belongs. M-dependent signals were treated in [6, 17] without assuming the boundedness of $\{X_n\}$. However, the moments of Y_n up to the order of 24 need to be bounded. The present work generalizes the previous results by relaxing both the boundedness assumption and the moment restriction.

Idea of the proof of Theorem 3.1: The proof is divided into two main steps. The first step is to show that (1.1) is bounded w.p.1. Rewrite (1.1) as

$$\begin{aligned} Z_{n+1} &= Z_n - a_nY_nY_n{}'Z_n + a_n(Y_n\psi_n - Y_nY_n{}'x^*) \\ &= \prod_{j=k+1}^{n}(I - a_jY_jY_j{}')Z_k + \sum_{j=k}^{n} a_j \prod_{l=j+1}^{n}(I - a_lY_lY_l')(Y_j\psi_j - Y_jY_j{}'x^*) \quad (3.1) \end{aligned}$$

where $Z_n = X_n - x^*$. In view of (A1) and (A2),

$$\sum_{j=1}^{\infty} a_j(Y_j\psi_j - Y_jY_j{}'x^*) \text{ converges w.p.1.} \tag{3.2}$$

Let λ be the smallest eigenvalue of R, and define $M(k,d) = \max\left\{n; \sum_{j=k}^{n} a_j \le d\right\}$.

Lemma 3.2. *There exist* $\eta > 0$ *with* $\eta\lambda < 1$, *such that for sufficiently large* k,

$$|\prod_{j=k}^{M(k,\eta)}(I - a_jY_jY_j{}')| \le 1 - \frac{1}{3}\eta\lambda.$$

Select η and k as in Lemma 3.2. For any $n \ge M(k,\eta)$, we divide $[k,n]$ into subintervals in the following way: $k = k_0 < M(k_0,\eta) = k_1 < \ldots < M(k_{r-2},\eta) = k_{r-1} < k_r = n$. Then, we work on these subintervals and proceed to prove the boundedness. The details are similar to that of [20, 26].

The second step is to establish the convergence properties. Define $V(z) = \frac{1}{2}z'z$. Due to (1.1), it is not hard to convince oneself that $V(Z_n) \neq$ constant. Suppose that the limit of Z_n does not exist. Then, the boundedness of Z_n would imply that there are two subsequences $\{Z^1_{n_k}\}$ and $\{Z^2_{n_k}\}$ such that $Z^1_{n_k} \to z_1$, $Z^2_{n_k} \to z_2$. Hence, $V(Z^2_{n_k}) - V(Z^1_{n_k}) \to V(z_2) - V(z_1) > 0$. Notice that for some $K > 0$, due to the boundedness of $\{Z_n\}$,

$$\begin{aligned} |V(Z_{n+1}) - V(Z_n)| &= |Z_{n+1}{}'Z_{n+1} - Z_n{}'Z_n| \\ &\le K(a_n|R||Z_n| + |a_n(Y_nY_n' - R)||Z_n| + |a_n(Y_n\psi_n - Y_nY_n'x^*)|). \end{aligned}$$

In view of the assumptions and the boundedness of Z_n, $V(Z_{n+1}) - V(Z_n) \xrightarrow{n} 0$. Thus $\{V(Z_n)\}$ would be across some interval $[\Delta_1, \Delta_2] \subset [V(z_1), V(z_2)]$ infinitely often.

By virtue of (A1)-(A2), for any $\tilde{\eta} > 0$, there exists a $K(\tilde{\eta}) > 0$ such that for all $l \ge 1$, $n \ge K(\tilde{\eta})$,

$$\begin{aligned} &|\sum_{j=n-1}^{n+l} a_j(Y_jY_j{}' - R)| \le \tilde{\eta}, \; |\sum_{j=n-1}^{n+l} a_j(Y_j\psi_j{}' - Q)| \le \tilde{\eta}, \\ &|\sum_{j=n-1}^{n+l} a_j(Y_j\psi_j{}' - Y_jY_j{}'x^*)| \le \tilde{\eta} \end{aligned} \tag{3.3}$$

Let μ, ν, with $K(\tilde{\eta}) \leq \mu \leq \nu$, be the first entrance from the left and first exit from the right of $\{V(Z_n)\}$ to $[\Delta_1, \Delta_2]$, respectively; that is,

$$\mu = \min\{n \geq K(\tilde{\eta});\ V(Z_n) \geq \Delta_1\}, \ \nu = \min\{n \geq K(\tilde{\eta});\ V(Z_n) \geq \Delta_2\}.$$

It then follows $V(Z_{\mu-1}) < \Delta_1 \leq V(Z_\mu) \leq \Delta_2 \leq V(Z_\nu)$.

In addition, it can be shown that there exists an $\varepsilon_0 > 0$ such that for all $0 \leq \varepsilon \leq \varepsilon_0$ and $l \in [\mu - 1, M(\mu - 1, \varepsilon) + 1]$, $|Z_l - Z_{\mu-1}| = O(\varepsilon)$.

Now, we have

$$\begin{aligned} &V(Z_{M(\mu-1,\varepsilon)+1}) - V(Z_{\mu-1}) \\ &= - Z'_{\mu-1} \sum_{j=\mu-1}^{M(\mu-1,\varepsilon)+1} a_j R Z_{\mu-1} + Z'_{\mu-1} a_{M(\mu-1,\varepsilon)+1} R Z_{\mu-1} \\ &- Z'_{\mu-1} \sum_{j=\mu-1}^{M(\mu-1,\varepsilon)} a_j (Y_j Y_j - R) Z_{\mu-1} + Z'_{\mu-1} \sum_{j=\mu-1}^{M(\mu-1,\varepsilon)} a_j (Y_j \psi_j - Y_j Y_j{}' x^*) \\ &- Z'_{\mu-1} \sum_{j=\mu-1}^{M(\mu-1,\varepsilon)} a_j Y_j Y_j (Z_j - Z_{\mu-1}) + O(\varepsilon^2). \end{aligned} \tag{3.4}$$

In view of (3.3), and choosing $\tilde{\eta} = o(\varepsilon)$, the middle two terms on the right-hand side of (3.4) are of the order $o(\varepsilon)$. As for the next to last term, the following estimate holds

$$\begin{aligned} &|Z'_{\mu-1} \sum_{j=\mu-1}^{M(\mu-1,\varepsilon)} a_j Y_j Y_j{}'(Z_j - Z_{\mu-1})| \\ &\leq K\varepsilon \sum_{j=\mu-1}^{M(\mu-1,\varepsilon)} a_j |Y_j|^2 \leq K\varepsilon \sum_{j=\mu-1}^{M(\mu-1,\varepsilon)} a_j tr Y_j Y_j{}' = o(\varepsilon). \end{aligned}$$

Thus, for some $\gamma > 0$,

$$V(Z_{M(\mu-1,\varepsilon)+1}) \leq V(Z_{\mu-1}) - \gamma\varepsilon + \gamma a_{M(\mu-1,\varepsilon)+1} + o(\varepsilon) < \Delta_1. \tag{3.5}$$

On the other hand, we can choose ε_0 so that $\mu - 1 < \mu \leq M(\mu - 1, \varepsilon) + 1 \leq \nu$ and hence, $\Delta_1 \leq V(Z_{M(\mu-1,\varepsilon)+1}) \leq \Delta_2$. This contradicts (3.5). Therefore, $\lim_n Z_n$ must exist.

Next, we show that $Z_n \xrightarrow{n} 0$ w.p.1. For suppose not, say $Z_n \xrightarrow{n} Z \neq 0$. Then, $V(Z_n) \xrightarrow{n} V(Z) \neq 0$. But, by using similar argument as before, and replacing $\mu - 1$ in (3.4) by n, we can show that $\lim_n V(Z_{M(n,\varepsilon)+1}) < V(Z) - \gamma_1\varepsilon$, for some $\gamma_1 > 0$. This is a contradiction. The proof of Theorem 3.1 is thus concluded.

4. Asymptotic distribution

Our task in this section is to show that $\sqrt{n}(X_{n+1} - x^*)$ has a normal limiting distribution. For simplicity, we shall assume $a_n = \frac{1}{n}$ henceforth. Similar results can be obtained for $a_n = A/n^\alpha$, with $0 < \alpha \leq 1$. In the subsequent development, we shall focus on the case of stationary φ-mixing type of signal models. Suppose that $\{\xi_n\}$ is a stationary process, and $\mathcal{F}_n$, $\mathcal{F}^n$ are σ-algebras given by $\mathcal{F}_n = \sigma\{\xi_k; k \leq n\}$ and $\mathcal{F}^n = \sigma\{\xi_k; k \geq n\}$. We say that $\{\xi_n\}$ is φ-mixing with mixing measure $\varphi_p(m)$ if for each n, m and $A \in \mathcal{F}^{n+m}$, $|P(A|\mathcal{F}_n) - P(A)| \leq \varphi_p(m)$, where $\varphi_p(m) = \sup_{A \in \mathcal{F}^{n+m}} E^{\frac{1}{p}} |P(A|\mathcal{F}_n) - P(A)|^p$. In particular, if $p = \infty$, then the mixing process is also called uniform mixing.

To proceed, we assume the following condition holds.

(A3) The assumption (A1) holds. In addition, $\{(Y_n, \psi_n)\}$ is a stationary φ-mixing sequence, such that

(i) $E|Y_n|^{4+\alpha} < \infty$, for some $\alpha > 0$.

(ii) $E|\psi_n|^{4+\beta} < \infty$ for some $\beta > 0$.

(iii) $I/2 - R$ is a stable matrix in that all eigenvalues of $I/2 - R$ have negative real parts.

(iv) Denote $\xi_n = Y_n \psi_n - Y_n {Y_n}' x^*$. For $m > 0$, let

$$\varphi(m) = \sup_{A \in \mathcal{F}^{n+m}} E^{\frac{1+\delta}{2+\delta}} [P(A|\mathcal{F}_n) - P(A)]^{\frac{2+\delta}{1+\delta}}, \tag{4.1}$$

where $\delta = \min(\frac{\alpha}{2}, \frac{\beta}{2})$. Suppose $\sum_m \varphi(m)^{\frac{\delta}{1+\delta}} < \infty$.

By using the functional invariance approach, a functional central limit theorem was also established independently in a recent work [9]. Among other things, an interesting law of iterated logarithm was obtained. Uniform mixing with mixing measure defined by the sup-norm is treated in [9], whereas φ-mixing with mixing measure defined by p-norm is considered in our paper. As far as the random signals are concerned, the conditions in our paper appear to be weaker.

It can be verified that under the mixing condition (A3), the condition (A2) is satisfied. Hence, the algorithm is convergent. In view of (i), (ii) and (iv), $\sup_n E|\xi_n|^{2+\delta} < \infty$.

Next, let

$$A_{jk} = \begin{cases} \prod_{l=k+1}^{j} (I - \frac{R}{l}), & j \geq k+1; \\ I, & j = k. \end{cases} \tag{4.2}$$

Lemma 4.1. *$\sqrt{n} Z_{n+1} = \sqrt{n} \sum_{k=1}^{n} \frac{1}{k} A_{nk} \xi_k + o(1)$ where $o(1) \xrightarrow{n} 0$ in probability.*

We omit the proof here. Owing to Lemma 4.1, to investigate the limiting distribution of $\sqrt{n}(X_{n+1} - x^*)$, we need only consider the quantity $\sqrt{n} \sum_{k=1}^{n} \frac{1}{k} A_{nk} \xi_k$. In order to study the asymptotic distribution, weak convergence methods seem to be extremely useful (cf. [2, 5, 11] and the references therein). For related work on invariance principles that emphasizes on the strong limit theory, we refer to the work [8] and the references therein.

To proceed, let $[z]$ denote the largest integral part of z. We define

$$M_n(t) = \frac{[nt]}{\sqrt{n}} \sum_{k=1}^{[nt]} \frac{1}{k} A_{[nt]k} \xi_k, \quad t \in [0,1], \; W_n(t) = \frac{1}{\sqrt{n}} \sum_{k=1}^{[nt]} \xi_k, \quad t \in [0,1]. \tag{4.3}$$

Rather than proving a central limit theorem for $\sqrt{n}(X_{n+1} - x^*)$, we derive a functional invariance theorem. This provides more information on the asymptotic behavior of the processes and is particularly useful when we want to deal with problems like random time change.

Lemma 4.2. *Under (A3), $W_n(\cdot)$ converges weakly to a Brownian motion with mean 0 and covariance*

$$S_0 = E(\xi_1 \xi_1') + \sum_{k=2}^{\infty} E(\xi_1 \xi_k') + \sum_{k=2}^{\infty} E(\xi_k \xi_1'). \tag{4.4}$$

Idea of proof: This can be proved similarly as that of Theorem 7.3.1 in [5].

Theorem 4.3. *Under the assumptions of the above lemma, $M_n(\cdot)$ converges weakly to $M(\cdot)$ which satisfies the following equation $M(t) = \int_0^t e^{-(I-R)(\ln u - \ln t)} dW(u)$ with $W(\cdot)$ given by Lemma 4.2.*

Remark: The notion of weak convergence is a substantial generalization of convergence in distribution. Due to the interpolations, $M_n(\cdot)$ and $W_n(\cdot)$ are in $D^r[0,1]$ the space of functions defined on the time interval $[0,1]$, that are right continuous and have left limits endowed with the Skorohod topology (cf. [2, 5, 11] and the references therein). It is easily seen that the limit process $M(\cdot)$ is a Gauss-Markov process.

We omit the proof here. A similar approach for some related problems can be found in [19]. Under the assumptions of the above theorem, $\sqrt{n}(X_{n+1} - x^*) \sim N(0, S)$, with S given by

$$S = \int_0^\infty e^{(\frac{1}{2}-R)u} S_0 e^{(\frac{1}{2}-R)' u} du$$

where S_0 is as in (4.4).

5. A stopping rule

Since the approximation scheme to the optimal weighting vector x^* is a recursive procedure, it will be very helpful if we can design a stopping rule. The stopping rule should be so designed that with desired accuracy, the iterates will terminate if the criterion is met. In an early paper [19], the stopping rule problem for stochastic approximation type algorithms was considered.

A good choice for the determination of stopping rules seems to be a confidence region approach. What we are seeking for is that $P\{x^* \in$ desired region$\}$ is near 1. Due to the nature of the algorithm, this kind of requirement is very hard to be fulfilled. We can, however, use an alternative–the asymptotic approach. Roughly speaking, the procedure can be described as follows. Picking out an $\varepsilon > 0$ and a confidence level α, we construct a sequence of ellipsoidal confidence regions such that as $\varepsilon \to 0$, $P\{x^*; x^* \in$ the confidence ellipsoid with volume $\leq \varepsilon^r\} \to 1 - \alpha$.

5.1. Ellipsoidal confidence regions. Since $\sqrt{n}(X_{n+1} - x^*) \sim N(0, S)$, if we can find a sequence S_n such that $S_n^{-1} \to S^{-1}$ in probability, then $n(X_{n+1} - x^*)' S_n^{-1} (X_{n+1} - x^*) \xrightarrow{n} \chi_r^2$ in distribution, where χ_r^2 denotes the Chi-square distribution with r degrees of freedom. Furthermore,

$$E^n = \{x^*;\ n(X_{n+1} - x^*)' S_n^{-1} (X_{n+1} - x^*) \leq c_\alpha\} \tag{5.1}$$

represents an ellipsoidal region and the volume of this region is given by

$$V(E^n) = \frac{\pi^{r/2} (c_\alpha / n)^{r/2} (\det S_n)^{1/2}}{\Gamma(1/2r + 1)} \tag{5.2}$$

where $\Gamma(\cdot)$ denotes the Γ-function.

In view of the above paragraph, for any $\varepsilon > 0$, we can choose $V(E^n) \leq \varepsilon^r$. Define

$$\nu_\varepsilon^n = \frac{\pi c_\alpha (\det S_n)^{1/r}}{\varepsilon^2 (\Gamma(1/2r + 1))^{2/r}},\ N_\varepsilon = \inf\{n; \nu_\varepsilon^n \leq n\}. \tag{5.3}$$

Define ν_ε as in ν_ε^n but with S_n replaced by S. N_ε is the stopping rule. The desired limit theorem reads:

Theorem 5.1. *If there exists a sequence $\{S_n\}$, such that $S_n \to S$ in probability with S given by (4.4), and if (A3) is satisfied, then*

$$\lim_{\varepsilon\to 0} \frac{N_\varepsilon}{\nu_\varepsilon} = 1 \text{ w.p.1 }, \ \lim_{\varepsilon\to 0} P\{x^* \in E^{N_\varepsilon} \text{ and } V(E^{N_\varepsilon}) \leq \varepsilon^r\} = 1-\alpha. \tag{5.4}$$

Idea of proof: By the definitions of ν_ε^n, ν_ε and N_ε, the first assertion above is easily verified. As for the second assertion, we need only show that the stopped process is asymptotically normal, or put it another way, the asymptotic normality holds with deterministic time n replaced by the stopping time N_ε. The proof of this fact depends on a random time change argument (cf. [2, 19]), and is in the same spirit of [19].

Define $\Phi_{\nu_\varepsilon}(t) = \begin{cases} tN_\varepsilon/\nu_\varepsilon, & \text{if } N_\varepsilon/\nu_\varepsilon \leq 1 \\ t; & \text{otherwise .}\end{cases}$ In view of (5.4), $\sup_{t\in[0,1]} |\Phi_{\nu_\varepsilon}(t) - t| \leq |N_\varepsilon/\nu_\varepsilon - 1| \to 0$ and $P\{N_\varepsilon/\nu_\varepsilon > 1\} \to 0$. By virtue of an argument as in [2], $M_{\nu_\varepsilon} \circ \Phi_{\nu_\varepsilon}$ converges weakly to $M \circ \Phi = M$ for all $t \in [0,1]$. Define $\tilde{M}_{\nu_\varepsilon}(t) = \frac{[N_\varepsilon t]}{\sqrt{\nu_\varepsilon}} \sum_{k=1}^{[N_\varepsilon t]} \frac{1}{k} A_{[N_\varepsilon t]k}\xi_k$. For $N_\varepsilon/\nu_\varepsilon \leq 1$, $M_{\nu_\varepsilon}(\Phi_{\nu_\varepsilon}(t)) = \tilde{M}_{\nu_\varepsilon}(t)$. It follows that $M_{\nu_\varepsilon}(\Phi_{\nu_\varepsilon}(\cdot))$ and $\tilde{M}_{\nu_\varepsilon}(\cdot)$ have the same weak limit. Thus, $\tilde{M}_{\nu_\varepsilon}(\cdot)$ converges weakly to $M(\cdot)$.

Since $|M_{N_\varepsilon}(t) - \tilde{M}_{\nu_\varepsilon}(t)| \leq |1 - \sqrt{\frac{\nu_\varepsilon}{N_\varepsilon}}||\tilde{M}_{\nu_\varepsilon}(t)|$, we have $M_{N_\varepsilon}(t) = \tilde{M}_{\nu_\varepsilon}(t) + o(1)$ where $o(1)\xrightarrow{\varepsilon}0$ in probability. Owing to the above equality, $M_{N_\varepsilon}(\cdot)$ and $\tilde{M}_{\nu_\varepsilon}(\cdot)$ have the same limit. As a result, $M_{N_\varepsilon}(\cdot)$ converges weakly to $M(\cdot)$. Similar to Lemma 4.2, it can be shown that $\sqrt{N_\varepsilon}(X_{N_\varepsilon+1} - x^*) = \sqrt{N_\varepsilon} \sum_{k=1}^{N_\varepsilon} \frac{1}{k} A_{N_\varepsilon,k}\xi_k + o(1)$, where $o(1)\xrightarrow{\varepsilon}0$ in probability. Thus, $\sqrt{N_\varepsilon}(X_{N_\varepsilon+1} - x^*)\xrightarrow{\varepsilon}N(0,S)$ in distribution.

5.2 Sequence of estimates. We demonstrate in this section that a consistent sequence S_n can be constructed. Define

$$\tilde{\xi}_n = Y_n\psi_n - Y_nY_n'X_n, \ S_n^i = \frac{1}{n}\sum_{j=1}^{n} \tilde{\xi}_j\tilde{\xi}'_{j+i}, \ i = 0, 1, ..., n-1, \ \tilde{S}_n = S_n^0 + \sum_{i=1}^{n-1}(S_n^i + S_n^{i,\prime}).$$

By the ergodicity, it is fairly easy to see that $\tilde{S}_n \xrightarrow{n} S$ in probability.

Next, define S_n by $S_n = \int_0^\infty e^{(I/2-S_n^0)t}\tilde{S}_n e^{(I/2-S_n^0)t}dt$. This definition is very explicitly. Nevertheless, as far as computation is concerned, such a formula may not be so attractive. However, S_n can be constructed by solving the well-known Liapunov equation $S_n(I/2-S_n^0)+(I/2-S_n^0)S_n = -\tilde{S}_n$ recursively. The development of the stopping rule is thus completed.

6. Asymptotic efficiency

In this section, we concern ourselves with the efficiency and optimal design issues. First, in lieu of using a scalar step-size sequence $\{a_n\}$, a sequence of matrices $\{\Gamma_n\}$ was utilized. Such adaptive algorithms involve the update of $\{x_n\}$ and a sequence of matrix-valued estimates. When $n \to \infty$, the iterates converge to the optimal weighting vector x^* and the asymptotic covariance matrix $\Sigma = R^{-1}SR^{-1}$ is the optimal one among all possible candidates. We begin with the following algorithm:

$$X_{n+1} = X_n - \frac{R}{n}Y_nY_n'X_n + \frac{R}{n}Y_n\psi_n \tag{6.1}$$

where $\Gamma \in \mathbb{R}^{r\times r}$ is a non-singular matrix. We use the asymptotic covariance matrix as a performance index. The matrix Γ is introduced as an adjustable matrix-valued parameter such that the asymptotic optimality can be achieved. To evaluate the asymptotic performance, consider the following optimization problem. Choose an appropriate Γ so that the asymptotic covariance matrix is minimized. Recall that if A and B are two symmetric positive definite matrices, then $A > B$ means that $A - B$ is positive definite.

Under suitable conditions (virtually those as in the previous sections), the algorithm (6.1) is strongly convergent, i.e., $X_n \xrightarrow{n} x^*$ w.p.1. Moreover, $\sqrt{n}(X_{n+1} - x^*) \sim N(0, \hat{\Sigma})$, with $\hat{\Sigma}$ given by

$$\hat{\Sigma} = \int_0^\infty e^{(\frac{I}{2} - \Gamma R)u} \Gamma S \Gamma' e^{(\frac{I}{2} - \Gamma R)' u} du, \tag{6.2}$$

where S is the error covariance as given in Section 4. It can be shown that the 'best' choice of Γ is $\Gamma = R^{-1}$. This leads to the best asymptotic covariance given by: $\Sigma^* = R^{-1} S R^{-1}$. In various applications, R is not explicitly known. In such cases, constructing a sequence of consistent estimates of Γ is necessary. Since the output signal $\{Y_n\}$ is at our disposal, we can construct a sequence of estimates R_n of R as follows:

$$R_n = \frac{1}{n} \sum_{k=1}^{n} Y_k Y_k' y_k'$$

and then let

$$\Gamma_n = R_n^{-1}. \tag{6.3}$$

We replace the fixed Γ in (6.1) by Γ_n and consider

$$X_{n+1} = X_n - \frac{\Gamma_n}{n} Y_n Y_n' X_n + \frac{\Gamma_n}{n} Y_n \psi_n \tag{6.4}$$

Even though by the matrix inversion lemma the above computation can be made recursive, the computation is very intensive.

Recently, a very interesting new approach was proposed by Polyak [18] for a recursive optimization algorithm of stochastic approximation type. The main idea comes from the key word '*averaging*'. By appropriately taking averages of the iterates, one comes up with better, or more efficient algorithms.

Motivated by the aforementioned ideas, we propose the following algorithm for the adaptive filtering problem [24]. Let the sequences $\{X_n\}$ and $\{\bar{X}_n\}$ be given by:

$$X_{n+1} = X_n + \frac{1}{n^\gamma} Y_n (\psi_n - Y_n' X_n), \text{ for some } \frac{1}{2} < \gamma < 1 \tag{6.5}$$

$$\bar{X}_{n+1} = \bar{X}_n - \frac{1}{n+1} \bar{X}_n + \frac{1}{n+1} X_{n+1}. \tag{6.6}$$

Note $\bar{X}_n = \frac{1}{n} \sum_{j=1}^{n} X_j$.

It should be pointed out that in [18], only i.i.d random variables were considered. It was noted there for moving average type of random sequences. A prewhitening filter might be needed. In fact, his technique does not seem to be able to treat correlated noise cases. In [24], we used a very different approach. By using direct computation, we showed that

$$\sqrt{n}(\bar{X}_{n+1} - x^*) = \frac{R^{-1}}{\sqrt{n}} (\sum_{j=1}^{n} \xi_j) + o(1),$$

where $o(1) \xrightarrow{n} 0$ in probability. Then, using similar idea as in Section 4, we obtain $\sqrt{n}(\bar{X}_{n+1} - x^*) \sim N(0, R^{-1} S R^{-1})$, which establishes the optimality.

Notice that in the above approach, no matrix-valued estimates are needed. It is conceivable this approach will become popular in the near future.

Since the scaling factor (such as $\sqrt{n}$ or $n^{\gamma/2}$) and the asymptotic covariance matrix are often taken together as a mean to evaluate the rate of convergence, the above paragraph indicates that the averaging approach produces the best possible rate of convergence.

7. Further discussions

In this paper, we developed some asymptotic properties of adaptive filtering procedures. As was mentioned at the beginning of the paper, the algorithm has a wide range of applications in stochastic programing, adaptive signal processing, and stochastic control arose from various science, engineering and economic systems. We have concentrated on the problem with decreasing step size. Similar analysis can be developed for the constant step size algorithm:

$$X_{n+1} = X_n - \varepsilon Y_n(Y_n' X_n - \psi_n).$$

In this case, the pertinent notion of convergence is in the sense of weak convergence.

In what follows, we mention some of the possible topics for further study. The investigation of some of the questions addressed here has already been set in motion.

1. In implementing adaptive procedures, projection or truncation algorithms are used quite often. To study projected version of the adaptive filtering algorithms has clear practical value.
2. Due to the fact, the actual signals are often subject to distortion or 'perturbation', one needs to study the robustness issue.
3. The underlying signals are often subject to time delay. The investigation of such problems seems to be necessary.

References

[1] A. Benveniste, M. Metivier and P. Priouret, *Algorithmes Adaptatifs Approximations Stochastiques*, Masson, Paris, 1987.

[2] P. Billingsley, *Convergence of Probability Measures*, J. Wiley, New York, 1968.

[3] H.F. Chen, *Recursive Estimation and Control for Stochastic Systems*, J. Wiley, New York, 1985.

[4] A.Z. Di, "An iterative algorithm for adaptive array processing with correlated and nonstationary signals", *J. Systems Sci. Math. Sci.* **3** (1983), 28-40.

[5] S. Either and T. Kurtz, *Markov Processes*, J. Wiley, New York, 1986.

[6] E. Eweda and O. Macchi, "Quadratic mean and almost sure convergence of unbounded stochastic approximation algorithm with correlated observations", *Ann. Inst. Henri Poincare, Section B* **19** (1983), 235-255.

[7] L. Gerencsér, "On a class of mixing processes", *Stochastics* **26** (1989), 165-191.

[8] C.C. Heyde, "Invariance Principles in Statistics", *Internat. Statist. Review* **49** (1981), 143-152.

[9] A. Heunis, "Rates of convergence for an adaptive filtering algorithm driven by stationary dependent data", to appear in *SIAM J. Control Optim.*

[10] H.J. Kushner, "A projected stochastic approximation method for adaptive filters and identifiers", *IEEE Trans. Automat. Control* **AC-25** (1980), 836-838.

[11] H.J. Kushner, *Approximation and Weak Convergence Methods for Random Processes with Applications to Stochastic Systems Theory*, MIT Press, Cambridge, 1984.

[12] H.J. Kushner and D. Clark, *Stochastic Approximation Method for Constrained and Unconstrained Systems*, Springer-Verlag, New York, 1978.

[13] H.J. Kushner and H. Huang, "Asymptotic properties of stochastic approximations with constant coefficients", *SIAM J. Control Optim.* **19** (1981), 87-105.

[14] H.J. Kushner and A. Shwartz, "Weak convergence and asymptotic properties of adaptive filters with constant gains", *IEEE Trans. Inform. Theory* **IT-30** (1984), 177-182.

[15] L.Ljung, "Analysis of recursive stochastic algorithms", *IEEE Trans. Automat. Control* **AC-22** (1977), 551-575.

[16] E. Lukacs, *Stochastic Convergence*, (Academic Press, New York, 1970).

[17] O. Macchi and E. Eweda, "Second order convergence analysis of stochastic adaptive linear filter", *IEEE Trans. Automat. Control* **AC-28** (1983), 76-86.

[18] B.T. Polyak, New method of stochastic approximation type, *Automat. Remote Control* **51** (1990), 937-946.

[19] G. Yin, "A stopping rule for the Robbins-Monro method", *J. Optim. Theory Appl.* **67** (1990), 151-173.

[20] G. Yin, "Asymptotic properties of an adaptive beam former algorithm", *IEEE Trans. Inform. Theory* **IT-35** (1989), 859-867.

[21] G. Yin, "Recent progress in parallel stochastic approximations", to appear in *Statistical Theory of Identification and Adaptive Control*, (P.E. Caines and L. Gerencsér Eds.), Springer-Verlag.

[22] G. Yin and Y.M. Zhu, "A robustness approach to adaptive filtering", *Proc. 28th Conference on Decision and Control*, (1989).

[23] G. Yin and Y.M. Zhu, "Time delay in adaptive filtering", *Proc. 29th Conference on Decision and Control*, (1990).

[24] G. Yin and Y.M. Zhu, "Averaging procedures in adaptive filtering: An efficient approach", preprint, (1991).

[25] B. Widrow and M. Hoff, Jr., "Adaptive switching circuits", in *IRE WESCON Conf. Rec.*, pt. 4, 96-104, 1960.

[26] Y.M. Zhu and G. Yin, "Adaptive filter with constraint and correlated nonstationary signals", *Systems Control Lett.* **10** (1988), 271-279.

ROBUST LMS ADAPTIVE FILTERS FOR NONGAUSSIAN NOISE ENVIRONMENTS: CONVERGENCE PROPERTIES[1]

Geoffrey A. Williamson Peter M. Clarkson

Department of Electrical and Computer Engineering
Illinois Institute of Technology
Chicago IL 60616

Gradient-descent based adaptive filters such as the Least Mean Square (LMS) algorithm descend a theoretical error surface by using "instantaneous gradient estimates," and implicitly averaging these estimates over successive iterations [4]. At convergence, an excess mean square error (MSE) cost is incurred because of noise in the gradient estimates. Only under very restrictive conditions (the primary example being that successive gradient estimates are independent gaussian samples) does the averaging in LMS provide a minimum variance gradient estimate, and hence minimize the excess MSE. In many situations, especially those involving impulsive and other nongaussian noise, the gradient noise distribution is such that other gradient estimators are more efficient. Furthermore, deviations from the assumed gradient noise distribution may occur, even when that distribution is gaussian. Thus, we seek gradient estimators for adaptive filters which, while preserving the simplicity of LMS, are tailored to, but are also *robust* to occasional and potentially severe departures from, an assumed noise distribution.

As an alternative to averaging the instantaneous gradient estimates, we consider the use of order statistic filters [1] on the last N instantaneous estimates. A variety of these order statistic filters, including the median and the trimmed mean, are in many situations robust estimators. We term the resulting generalization of the LMS algorithm an Order Statistic LMS (OSLMS) algorithm. This idea has been shown to be effective for adaptive filtering in impulsive noise environments when using a median filter on the instantaneous estimates [2, 5, 6]. The use of trimmed mean filters has been shown to have superior performance (relative to LMS) in noise environments ranging between impulsive and gaussian [3]. The trimmed mean can be viewed as a compromise between raw averaging and the sample median, with the trimming factor in the filter adjusted to match the expected noise environment in an effort to enhance performance.

In this paper we address convergence issues for a class of OSLMS algorithms which includes those using median and trimmed mean gradient estimators. We demonstrate a convergence result for this class when the filter inputs are iid, zero mean symmetric.

The general OSLMS algorithm with filter coefficients $\underline{f}_k$, data vector $\underline{x}_k$, and prediction error ϵ_k is given as

$$\underline{f}_{k+1} = \underline{f}_k + \mu \mathbf{OS}_{\underline{a}}\{\underline{x}_k \epsilon_k\}_N. \tag{1}$$

The instantaneous gradient estimates of LMS, $\underline{x}_k \epsilon_k$, are passed through an order statistic filter $\mathbf{OS}_{\underline{a}}\{\cdot\}_N$, indexed by a weight vector $\underline{a}$, before updating the filter coefficients. For a scalar sequence s_k, the output of such a filter is

$$\mathbf{OS}_{\underline{a}}\{s_k\}_N = \left[\begin{array}{ccc} s_{k,(1)} & \cdots & s_{k,(N)} \end{array}\right] \underline{a},$$

where $s_{k,(r)}$ is the rth ranked value of $\{s_k, \ldots, s_{k-N+1}\}$ (ranked lowest to highest), and the vector operation is defined component-by-component. We restrict consideration here to $\underline{a}$ satisfying

---

[1] Supported by the National Science Foundation under grant nos. MIP-8911676 and MIP-9102620.

- $a(i) \geq 0$ ($a(i)$ denotes the ith element of $\underline{a}$)
- $\sum_i a(i) = 1$
- $a(r) = a(N - r + 1)$.

These conditions on $\underline{a}$ define OS filters $\mathbf{OS}_{\underline{a}}\{\underline{s}_k\}_N$ which are symmetric convex combinations of the ranked components of $\{\underline{s}_k, \cdots, \underline{s}_{k-N+1}\}$. Trimmed mean OS filters are special members of this class with $\underline{a} = [0 \cdots 0 \ \underbrace{1/T \cdots 1/T}_{T \text{ entries}} \ 0 \cdots 0]^t$. This $\underline{a}$ defines the median with $T = 1$, and the average with $T = N$.

Make the following assumptions:

1. The filter inputs $\{x_k\}$ are iid samples drawn from a zero mean symmetric density,
2. $\underline{x}_k \epsilon_k$ is independent of $\underline{x}_\ell \epsilon_\ell$ for all $k \neq \ell$,
3. $\epsilon_k = \underline{x}_k^t(\underline{f}^* - \underline{f}_k^{av})$.

We then have our main result: for small μ, the average approximation to (1), given by

$$\underline{f}_{k+1}^{av} = \underline{f}_k^{av} + \mu E\left\{\mathbf{OS}_{\underline{a}}\{\underline{x}_k \epsilon_k\}_N \mid \underline{f}_k^{av}\right\}, \tag{2}$$

satisfies $\|\underline{f}^* - \underline{f}_{k+1}^{av}\|_\infty \leq \|\underline{f}^* - \underline{f}_k^{av}\|_\infty$. With a slow adaptation rate (small μ), (2) closely approximates the average behavior of (1). Hence, OSLMS on average is nondivergent. Simulation evidence shows that, given sufficient excitation, OSLMS in fact converges and does so exponentially fast. Note that Assumption 2 above is commonly employed in the analysis of adaptive algorithms. Though it is seldom precisely true, analyses using the assumption have provided reliable predictions of algorithm behavior. Assumption 3 indicates operation of the adaptive filter in the noiseless case. Though we do not show in this paper that the filter converges when noise is present, simulation evidence suggests that the mean convergence properties of the noiseless case continue to hold.

References

[1] A.C. Bovik, T.S. Huang, and D.C. Munson, Jr., "A generalization of median filtering using linear combinations of order statistics," *IEEE Trans. on Acoust., Speech, and Sig. Proc.*, vol. ASSP-31, pp. 1342-1350, Dec. 1983.

[2] P.M. Clarkson and T.I. Haweel, "A median adaptive filter," *Electr. Letts.*, vol. 25, pp. 520-522, 1989.

[3] T.I. Haweel and P.M. Clarkson, "A class of order statistic LMS algorithms," *IEEE Trans. on Sig. Proc.*, to appear, Jan. 1992.

[4] B. Widrow and S. Stearns, *Adaptive Signal Processing.* Englewood Cliffs, NJ: Prentice-Hall, 1985.

[5] G.A. Williamson and P.M. Clarkson, "On convergence and steady state behavior in the median LMS adaptive filter," *Proc. ICASSP 91*, Toronto, Canada, May 1991, pp. 2149-2152.

[6] G.A. Williamson, P.M. Clarkson, and W.A. Sethares, "Performance characteristics of the median LMS adaptive filter," *IEEE Trans. on Sig. Proc.*, accepted for publication.

WAVELET DECOMPOSITION OF HARMONIZABLE RANDOM PROCESSES[1]

Ping Wah Wong
Department of Electrical and Computer Engineering
Clarkson University
Potsdam, NY 13699

Abstract

We consider the discrete wavelet decomposition of general harmonizable random processes. As in the case for square integrable deterministic functions, there are two different but equivalent forms in representing the discrete wavelet decomposition of processes where the spectral measures have no point mass at the frequency origin. For general harmonizable processes, however, only one of these forms is valid in providing a multiresolution decomposition. The harmonizability of wavelet decompositions is also considered.

1 Introduction

Continuous wavelet transform is a technique for decomposing a signal into components that have good localization properties both in time and frequency [1, 2, 3]. Because of the time and frequency localization properties, wavelet transform can provide local information on a signal that cannot be obtained using, for example, the Fourier transform. Mallat introduced the discrete wavelet decomposition of square integrable signals, *i.e.*, those that are in L_2, and developed a technique of multiresolution signal analysis using wavelets [4, 5]. On the other hand, the discrete wavelet decomposition can also be viewed as an orthonormal decomposition of L_2.

Since many useful and interesting signals encountered in practice are best modeled as random processes, a theory of wavelet decomposition for general random processes is needed. Treatments in this regard have so far been reported for nearly $1/f$ processes [6], fractional Brownian motion [7, 8], finite energy and truncated (on a finite subset of the real line) finite power processes [9]. In this summary, we consider the wavelet decomposition of harmonizable processes [10, Section 37.4]. Roughly speaking, a harmonizable process is a second-order process that can be represented as a superposition of complex exponentials. In general, a harmonizable process can either be wide-sense stationary or

---

[1] This work was supported in part by NSF grant MIP-9108410.

non-stationary. In the special case where it is wide-sense stationary, then the harmonic decomposition is identical to the well known spectral representation theorem of Bochner [11, Sections 20-21]. We first describe in the following the two different forms of the wavelet decomposition of deterministic functions outside the space L_2 [12]. We then present the new results on the wavelet decomposition of harmonizable processes. Analogous to the deterministic case, we shall see that the stochastic wavelet decomposition can also be represented in two different forms. They are equivalent for processes where the spectral measure has no point mass at the origin (zero frequency). Only one of these, however, is valid for the decomposition of general harmonizable processes.

2 Deterministic Wavelet Decomposition

The discrete wavelet transform pair for a deterministic square integrable function $x(t)$, denoted by $x(t) \in L_2$, is [4, 5, 13]

$$X_{m,n} = \int_{-\infty}^{\infty} x(t)\overline{\psi}_{m,n}(t)\, dt \tag{1}$$

and

$$x(t) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} X_{m,n}\psi_{m,n}(t), \tag{2}$$

where the overbar denotes complex conjugate, and $\psi_{m,n}(t)$ is a dilated and translated version of a *wavelet function* $\psi(t) \in L_2$, *viz.*,

$$\psi_{m,n}(t) = 2^{m/2}\psi(2^m t - n).$$

Associated with each $\psi(t)$, there is a *scaling function* $\phi(t)$, also in L_2, that satisfies the relationship

$$x_L(t) \triangleq \sum_{n=-\infty}^{\infty} S_{L,n}\phi_{L,n}(t) = \sum_{m=-\infty}^{L-1} \sum_{n=-\infty}^{\infty} X_{m,n}\psi_{m,n}(t), \tag{3}$$

where $\phi_{m,n}(t)$ is defined similarly as in $\psi_{m,n}(t)$, and

$$S_{L,n} = \int_{-\infty}^{\infty} x(t)\overline{\phi}_{L,n}(t)\, dt. \tag{4}$$

In the language of multiresolution analysis [4], $x_L(t)$ can be interpreted as the approximation of $x(t)$ up to the resolution level L. For any $x(t) \in L_2$, $x_L(t)$ satisfies

$$\lim_{L\to-\infty} x_L(t) = 0 \tag{5}$$

and

$$\lim_{L\to\infty} x_L(t) = x(t).$$

Because of (3), $x(t)$ can also be written as

$$x(t) = \sum_{m=K}^{\infty} \sum_{n=-\infty}^{\infty} X_{m,n}\psi_{m,n}(t) + \sum_{n=-\infty}^{\infty} S_{K,n}\phi_{K,n}(t) \qquad \text{any finite } K. \tag{6}$$

For any $x(t) \in L_2$, the representations (2) and (6) are completely equivalent in the sense that the right sides of both equations converge to $x(t)$. This is not the case for functions outside of L_2 [12].

In order to define the wavelet decomposition for non-square integrable functions, we need to impose more restrictive conditions on both $\psi(\cdot)$ and $\phi(\cdot)$ so that the coefficients $X_{m,n}$'s and $S_{K,n}$'s in (1) and (4) are well defined, *i.e.*, so that the integrals converge, and that the decomposition converges to the function. This is commonly referred to as a regularity condition. Here, we impose a regularity condition used by Mallat.

Definition: A wavelet function $\psi(\cdot)$ is said to be regular [4, 5] if and only if it is continuously differentiable, and also satisfies

$$\left|\frac{d^q\psi(t)}{dt^q}\right| \le \frac{D_q}{(1+|t|)^2} \qquad \forall t; \quad q = 0, 1, \tag{7}$$

where D_0 and D_1 are finite constants. The regularity of $\phi(\cdot)$ is defined in the same way. ∎

Under such condition, (6) is valid when the wavelet coefficients exist. Equation (2), however, fails to hold unless $x(t)$ is square integrable. For example, if $x(t) = 1$ for all t, then $X_{m,n}(t) = 0$ for all m and n, and hence (2) is false. The reason for this is that (5) no longer holds for functions outside of L_2 [12] and hence (6) and (2) are not equivalent.

In the following, we state without proof two results which will be useful in establishing the wavelet decomposition of harmonizable processes.

Lemma 1: Let $\psi(\cdot)$ and $\phi(\cdot)$ be regular wavelet and scaling functions, respectively. Their Fourier transforms are bounded by

$$\left|\widehat{\psi}(f)\right| \le \min\left(C_0, \frac{C_1}{|f|}, C_2|f|\right)$$

$$\left|\widehat{\phi}(f)\right| \le \min\left(C_3, \frac{C_4}{|f|}, 1 + C_5|f|\right)$$

where C_i's are finite constants. ∎

Lemma 2: Let $\psi(\cdot)$ be a regular wavelet function and $\phi(\cdot)$ be its corresponding regular scaling function. The wavelet decomposition of $e^{j2\pi f_0 t}$ is given by

$$e^{j2\pi f_0 t} = \lim_{M,N\to\infty} I_{K,M,N}(f_0, t) \qquad \text{any fixed finite } K$$

where

$$I_{K,M,N}(f,t) = \sum_{m=K}^{M} \sum_{n=-N}^{N} \overline{\hat{\psi}}(2^{-m}f)e^{j2\pi f 2^{-m}n}\psi(2^m t-n) + \sum_{n=-N}^{N} \overline{\hat{\phi}}(2^{-K}f)e^{j2\pi f 2^{-K}n}\phi(2^K t-n).$$

Furthermore, the convergence is uniformly bounded for all f_0 and t, *i.e.*, $I_{K,M,N}(f_0,t)$ is uniformly bounded for all $M > K$ and all N, f_0, t. ■

3 Decomposition of Harmonizable Processes

A process $x(t)$ is said to be of second-order if $E[x^2(t)] < \infty$ for all t [14, 10]. A large class of second-order processes are harmonizable, which, roughly speaking, are those that can be represented as a superposition of complex sinusoids. A precise definition of harmonizable processes due to Loève is given as follows:

Definition: [10, Section 37.4] A covariance function $R(t,t')$ is said to be *harmonizable* if there exists a covariance $\gamma(f,f')$ of bounded variation on $\mathbb{R} \times \mathbb{R}$ such that

$$R(t,t') = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{j2\pi(ft-f't')}\, dd\gamma(f,f'). \tag{8}$$

The symbol $\mathbb{R}$ denotes the set of real numbers. ■

Definition: [10, Section 37.4] A second-order random process $x(t)$ is said to be *harmonizable* if there exists a second-order process $\xi(f)$ with a covariance function $E[\xi(f)\overline{\xi}(f')] = \gamma(f,f')$ of bounded variation on $\mathbb{R} \times \mathbb{R}$ such that

$$x(t) = \int_{-\infty}^{\infty} e^{j2\pi ft}\, d\xi(f) \tag{9}$$

with probability one. ■

In the special case where $x(t)$ is also wide-sense stationary, the process $\xi(f)$, usually called the spectral process, has orthogonal increments, *i.e.*,

$$E[d\xi(t)d\overline{\xi}(s)] = dF(s)\,\delta_{t-s},$$

where δ_t is the Kronecker delta, and $F(\cdot)$, usually called the spectral measure, is a finite measure on the real line. Equation (8) for wide-sense stationary processes then becomes

$$E[x(t)\overline{x}(t')] = R_x(t-t') = \int_{-\infty}^{\infty} e^{j2\pi f(t-t')}dF(f),$$

which is the well known theorem of Bochner for the spectral decomposition of mean square continuous wide-sense stationary random processes [11, Sections 20-21].

Theorem 1: A second-order harmonizable random process $x(t)$ can be decomposed as

$$x(t) = \underset{M,N\to\infty}{\text{l.i.m.}} \left[\sum_{m=K}^{M} \sum_{n=-N}^{N} X_{m,n}\psi_{m,n}(t) + \sum_{n=-N}^{N} S_{K,n}\phi_{K,n}(t) \right] \qquad \text{any finite } K, \text{ any } t \tag{10}$$

where l.i.m. denotes limit in the mean square,

$$\begin{aligned} X_{m,n} &\triangleq \int_{-\infty}^{\infty} x(t)\overline{\psi}_{m,n}(t)\, dt \\ &= \int_{-\infty}^{\infty} 2^{-m/2}\overline{\hat{\psi}}(2^{-m}f)e^{j2\pi f 2^{-m}n}\, d\xi(f) \qquad m = K, K+1, \ldots, \end{aligned} \tag{11}$$

and

$$S_{K,n} \triangleq \int_{-\infty}^{\infty} x(t)\overline{\phi}_{K,n}(t)\, dt = \int_{-\infty}^{\infty} 2^{-K/2}\overline{\hat{\phi}}(2^{-K}f)e^{j2\pi f 2^{-K}n}\, d\xi(f). \tag{12}$$

Furthermore, the convergence in (10) is uniformly bounded in the mean square sense.

Proof: Using Lemma 1 and the fact that $\gamma(f, f')$ is of bounded variation, it can be verified that $X_{m,n}$'s and $S_{K,n}$'s are well defined. We can then write

$$E\left| x(t) - \sum_{m=K}^{M} \sum_{n=-N}^{N} X_{m,n}\psi_{m,n}(t) + \sum_{n=-N}^{N} S_{K,n}\phi_{K,n}(t) \right|^2$$

$$= E\left| \int_{-\infty}^{\infty} \left(e^{j2\pi ft} - I_{K,M,N}(f,t) \right) d\xi(f) \right|^2$$

$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \left(e^{j2\pi ft} - I_{K,M,N}(f,t) \right) \left(e^{j2\pi f't} - I_{K,M,N}(f',t) \right) dd\gamma(f,f').$$

We have interchanged the integration and expectation because the integrand is uniformly bounded and $\gamma(f, f')$ is of bounded variation. Furthermore we can apply the dominated convergence theorem to conclude that the limit is zero, proving (10). The fact that the convergence is uniformly bounded follows from Lemma 2. ∎

Equation (10) is said to be the *discrete wavelet decomposition of second-order harmonizable processes.* It is formally identical to (6) except that the equalities in the two cases must be interpreted in different senses. For each fixed m, $X_{m,n}$ and $S_{m,n}$ constitute two discrete time random processes. In the actual computation of the wavelet coefficients, one can apply as in the deterministic case the pyramidal algorithm of Mallat [4]. That is, given the process $S_{L,n}$ for a certain L, we can compute $S_{L-1,n}$'s and $X_{L-1,n}$'s by passing $S_{L,n}$ through digital filters followed by down-samplers. This procedure can be repeated recursively to compute all the wavelet coefficients $X_{m,n}$ for $m < L$.

Since (10) is valid for any finite K, it can be easily shown that

$$\sum_{n=-\infty}^{\infty} S_{L,n}\phi_{L,n}(t) = \sum_{n=-\infty}^{\infty} S_{L-1,n}\phi_{L-1,n}(t) + \sum_{n=-\infty}^{\infty} X_{L-1,n}\psi_{L-1,n}(t) \tag{13}$$

holds in the mean square sense for any finite integer L. This implies that if a process $x(t)$ with the corresponding $S_{K,n}$'s satisfy

$$\underset{K\to-\infty, N\to\infty}{\mathrm{l.i.m.}} \sum_{n=-N}^{N} S_{K,n}\phi_{K,n}(t) = 0, \tag{14}$$

then an alternative form of wavelet decomposition corresponding to (2) of the deterministic case will hold.

Theorem 2: Let $x(t)$ be a second-order wide-sense stationary process. Assume that its spectral measure is absolutely continuous at a neighborhood of zero frequency, then the equality

$$\sum_{n=-\infty}^{\infty} S_{L,n}\phi_{L,n}(t) = \sum_{m=-\infty}^{L-1} \sum_{n=-\infty}^{\infty} X_{m,n}\psi_{m,n}(t)$$

holds in the mean square sense for any finite integer L and any t. ∎

Proof: It suffices to show (14) is valid. To do so, consider

$$\lim_{K\to-\infty} \lim_{N\to\infty} E\left|\sum_{n=-N}^{N} S_{K,n}\phi_{K,n}(t)\right|^2$$

$$= \lim_{K\to-\infty} \sum_{n=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} a_{K,n,l}\phi(2^K t - n)\overline{\phi}(2^K t - l) \tag{15}$$

where

$$a_{K,n,l} = \int_{-\infty}^{\infty} \left|\widehat{\phi}(2^{-K} f)\right|^2 e^{j2\pi f 2^{-K}(n-l)} dF(f).$$

Note that $a_{K,n,l}$ is bounded independently of K, n and l because $\widehat{\phi}(\cdot)$ is bounded and $F(\cdot)$ is of bounded variation. We can then apply the dominated convergence theorem to interchange the order of the limit and the infinite sums in (15), and hence we need to show that $a_{K,n,l} \to 0$ as $K \to -\infty$. To this end, choose and fix a small enough ϵ so that $dF(f) \le C\, df$ for some constant C. This is possible because of the assumption that F is absolutely continuous at the origin. Letting $\epsilon_K = \epsilon/|K|$ for $K = -1, -2, \ldots$, we have

$$\begin{aligned} |a_{K,n,l}| &\le \int_{|f|\le\epsilon_K} \left|\widehat{\phi}(2^{-K} f)\right|^2 dF(f) + \int_{|f|>\epsilon_K} \left|\widehat{\phi}(2^{-K} f)\right|^2 dF(f) \\ &\le \int_{|f|\le\epsilon_K} C_3 C\, df + \int_{|f|\ge\epsilon_K} C_4^2 2^{2K} f^{-2}\, dF(f) \\ &\le \frac{2C_3 C\epsilon}{|K|} + \frac{C_4^2 2^{2K} K^2}{\epsilon^2} \int_{-\infty}^{\infty} dF(f). \end{aligned}$$

Hence $a_{K,n,l} \to 0$ as $K \to -\infty$ and the theorem is proved. ∎

Theorems 1 and 2 imply an alternative form of wavelet decomposition for a subclass of harmonizable processes as follows:

Theorem 3: If $x(t)$ is a second-order wide-sense stationary process where the spectral measure is absolutely continuous at a neighborhood of zero frequency, then its wavelet decomposition can be represented as

$$x(t) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} X_{m,n}\psi_{m,n}(t), \tag{16}$$

where $X_{m,n}$ is defined in (11). The equality is in the mean square sense, and the convergence in (16) is also uniformly bounded in the mean square sense. ∎

Note that both Theorems 2 and 3 can be strengthened. In particular, the requirement that the spectral measure be absolutely continuous at the origin can be weakened, and wide sense stationarity can be dropped. In other words, these two theorems hold if the spectral measure $\gamma(f, f')$ has no point mass at the origin [15]. It is easy to see that Theorem 2 fails to be true if the spectral measure has a point mass at dc. For example, if $x(t) = 1$ for all t, then $X_{m,n} = 0$ for all m, n while $S_{K,n} = 1$ for all n and for any fixed K. Roughly speaking, this is because $\widehat{\psi}_{m,n}(f)$ equals zero at $f = 0$ for any m and n. Hence, no linear combination of $\psi_{m,n}(t)$ can represent a process that has a point mass at zero frequency. In such case we must use a wavelet decomposition of the form (10). The next result concerns the wavelet decomposition of the covariance functions of harmonizable processes.

Theorem 4: The wavelet decomposition of a second-order process $x(t)$ exists if and only if its covariance function has the decomposition

$$\begin{aligned} R(t,t') = & \sum_{m,m'\geq K} \sum_{n,n'} \alpha_{m,m',n,n'}\psi_{m,n}(t)\overline{\psi}_{m',n'}(t') + \sum_{m\geq K} \sum_{n,n'} \eta_{m,K,n,n'}\psi_{m,n}(t)\overline{\phi}_{K,n'}(t') \\ & + \sum_{m'\geq K} \sum_{n,n'} \overline{\eta}_{m',K,n',n}\overline{\psi}_{m',n'}(t')\phi_{K,n}(t) + \sum_{n,n'} \beta_{K,n,n'}\phi_{K,n}(t)\overline{\phi}_{K,n'}(t'). \end{aligned} \tag{17}$$

When they exist, the coefficients are related by

$$\alpha_{m,m',n,n'} = E[X_{m,n}\overline{X}_{m',n'}],$$

$$\eta_{m,K,n,n'} = E[X_{m,n}\overline{S}_{K,n'}]$$

and

$$\beta_{K,n,n'} = E[S_{K,n}\overline{S}_{K,n'}].$$

∎

This theorem is essentially a consequence of the convergence in mean square criterion [10, page 135-6]. From the recursive decomposition algorithm of Mallat [4], the wavelet decomposition can be interpreted as a multistep linear filtering and down-sampling procedure on the original process. Hence if a process is harmonizable, *i.e.*, if it can be represented as a superposition of sinusoids, the output of such a process through linear filters and down-samplers should not change this property. The following result states this in a precise form.

Theorem 5: Let $x(t)$ be a second-order continuous random process, and let its wavelet decomposition be formally given by (10). Let

$$x_L(t) \triangleq \sum_{n=-\infty}^{\infty} X_{m,n}\psi_{L,n}(t)$$

and

$$s_L(t) \triangleq \sum_{n=-\infty}^{\infty} S_{L,n}\phi_{L,n}(t).$$

Then, the following statements are true:

1. If $x(t)$ is harmonizable, then $x_L(t)$ and $s_L(t)$ are also harmonizable for any L.
2. If for any fixed K, $s_K(t)$ is harmonizable, then $x_L(t)$ and $s_L(t)$ for all $L < K$ are harmonizable.
3. If for any fixed K, $s_K(t)$ is harmonizable, and $x_L(t)$'s are harmonizable for all $L \geq K$, then $x(t)$ is harmonizable.

■

Proof: First, consider statement 1. If $x(t)$ is harmonizable, then we can use (11) to write

$$x_L(t) = \sum_{n=-\infty}^{\infty} \int_{-\infty}^{\infty} \overline{\widehat{\psi}}(2^{-m} f) e^{j2\pi f 2^{-m} n}\, d\xi(f)\, \psi(2^m t - n).$$

It can be verified using the regularity of $\psi(\cdot)$ and the Fubini's theorem that we can interchange the order of integration and summation. Then, we can apply the Poisson sum formula to write

$$x_L(t) = \int_{-\infty}^{\infty} e^{j2\pi f t}\overline{\widehat{\psi}}(2^{-m} f) \left(\sum_{k=-\infty}^{\infty} \widehat{\psi}(2^{-m} f - k) e^{-j2\pi f 2^m k} \right) d\xi(f),$$

and hence $x_L(t)$ is harmonizable. Similarly, $s_L(t)$ is also harmonizable. Statement 2 follows from similar steps as above and the fact that

$$X_{m,n} = \int_{-\infty}^{\infty} x(t)\overline{\psi}_{m,n}(t)\, dt = \int_{-\infty}^{\infty} s_K(t)\overline{\psi}_{m,n}(t)\, dt \qquad m < K,$$

$$S_{m,n} = \int_{-\infty}^{\infty} x(t)\overline{\phi}_{m,n}(t)\, dt = \int_{-\infty}^{\infty} s_K(t)\overline{\phi}_{m,n}(t)\, dt \qquad m < K.$$

To prove statement 3, let

$$x_L(t) = \int_{-\infty}^{\infty} e^{j2\pi ft}\, d\xi_L(f) \qquad L = K, K+1, \ldots,$$

and

$$s_K(t) = \int_{-\infty}^{\infty} e^{j2\pi ft}\, d\rho_K(f),$$

where $\xi_L(f)$'s and $\rho_K(f)$ are the spectral processes. Because of (13), and the fact that the sum of two harmonizable processes are harmonizable, we can write

$$s_{L+1}(t) = \int_{-\infty}^{\infty} e^{j2\pi ft}\, d\rho_{L+1}(f)$$

where

$$\rho_{L+1}(f) = \rho_L(f) + \xi_L(f).$$

Letting $E[\rho_L(f)\overline{\rho}_L(f')] = \gamma_L(f, f')$, we want to show that

$$E[s_L(t)\overline{s}_L(t')] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{j2\pi(ft-f't')}\, dd\gamma_L(f, f') \tag{18}$$

converges to a harmonizable covariance function. From Theorem 1, $s_L(t)$ converges boundedly to $x(t)$ in mean square. Hence, (18) is uniformly bounded for all L and t. Then it follows from a convergence theorem of Stieltjes integrals [11, page 322] that

$$\lim_{L\to\infty} E[s_L(t)\overline{s}_L(t')] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{j2\pi(ft-f't')}\, dd\gamma(f, f')$$

for some measure $\gamma(f, f')$. Using the harmonizable theorem of Loève [10, page 142], we can conclude that $x(t)$ is harmonizable. ∎

References

[1] A. Grossmann and J. Morlet, "Decomposition of Hardy functions into square integrable wavelets of constant shape," *SIAM Journal of Mathematical Analysis*, vol. 15, pp. 723–736, July 1984.

[2] A. Grossmann, R. Kronland-Martinet, and J. Morlet, "Reading and understanding continuous wavelet transforms," in *Wavelets: Time-Frequency Methods and Phase Space, Proceedings of the International Conference* (J. M. Combes, A. Grossmann, and P. Tchamitchian, eds.), (Marseille, France), pp. 2–20, Springer-Verlag, December 1987.

[3] I. Daubechies, "The wavelet transform, time-frequency localization and signal analysis," *IEEE Transactions on Information Theory*, vol. IT-36, pp. 961–1005, September 1990.

[4] S. G. Mallat, "A theory for multiresolution signal decomposition: The wavelet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-11, pp. 674–693, July 1989.

[5] S. G. Mallat, "Multiresolution approximations and wavelet orthonormal bases of $L^2(R)$," *Transactions of The American Mathematical Society*, vol. 315, pp. 69–87, September 1989.

[6] G. W. Wornell, "A Karhunen-Loève-like expansion for $1/f$ processes via wavelets," *IEEE Transactions on Information Theory*, vol. IT-36, pp. 859–861, July 1990.

[7] A. H. Tewfik and M. Kim, "Correlation structure of the discrete wavelet coefficients of fractional Brownian motion," 1990. Preprint.

[8] P. Flandrin, "Wavelet analysis and synthesis of fractional brownian motion," 1990. Preprint.

[9] M. J. Genossar, M. Goldburg, H. Lev-Ari, and T. Kailath, "Extending wavelet decompositions to random processes with applications to periodically correlated processes," 1990. Preprint.

[10] M. Loève, *Probability Theory II.* New York: Springer-Verlag, 4th ed., 1977.

[11] S. Bochner, *Lectures on Fourier Integrals.* Princeton, NJ: Princeton University Press, 1959. Translated from the original by M. Tenenbaum and H. Pollard.

[12] Y. Meyer, "Orthonormal wavelets," in *Wavelets: Time-Frequency Methods and Phase Space, Proceedings of the International Conference* (J. M. Combes, A. Grossmann, and P. Tchamitchian, eds.), (Marseille, France), pp. 21–37, Springer-Verlag, December 1987.

[13] I. Daubechies, "Orthonormal bases of compactly supported wavelets," *Communications on Pure and Applied Mathematics*, vol. 41, pp. 909–996, November 1988.

[14] E. Wong, *Stochastic Processes in Information and Dynamical Systems.* New York: McGraw-Hill, 1971.

[15] M. Genossar, Private communication, 1991.

2-D ZERO PHASE RATIONAL FILTERS: DESIGN AND BOUNDARY-VALUED REALIZATION*

T.S. Rao and R.M. Mersereau
Digital Signal Processing Laboratory
School of Electrical Engineering
Georgia Institute of Technology
Atlanta, GA 30332
(404) 894-2910

Abstract

Zero-phase filters are useful in image processing applications in part because they perform the required frequency shaping without unduly distorting the lines, spots and gratings that are often found in images. In this paper we investigate the design and realization of 2-D zero-phase rational filters. A design algorithm, based on the McClellan transformation of a 1-D zero-phase IIR filter, is proposed and examples illustrating its simplicity and functionality are given. While the technique is integral to the topic at hand, the main focus and contribution of the paper lies with the filter's eventual realization. The novelty of our approach is that the designed filter is realized, not recursively using initial conditions, but simultaneously through the assumption of boundary conditions. The effect is to transform the cumbersome filtering task into a computationally tractable boundary value problem. Our results indicate that, in most instances, the boundary-valued system retains the frequency shaping characteristics of the underlying shift-invariant system.

---

*This research was supported in part by the Joint Services Electronics Program under Contract DAAL-03-90-C-0004 and in part by the GTRI/OOD GRA Support Program.

1 Introduction

Zero-phase filters constitute an important class of 2-D systems. These systems, characterized by symmetric impulse responses, are particularly useful in image processing applications in part because they perform the required frequency shaping without unduly distorting the lines, spots, and gratings that are often found in images.

While the design and implementation of 2-D zero-phase FIR filters has received considerable attention in recent years [1, 2, 3, 4, 5], the design and realization of 2-D zero-phase rational filters has gone largely ignored. Although there are several reasons for this, the main reason stems from the fact that it is impossible for a single recursively computable IIR filter to have zero-phase. Consequently, to date, the most practical way of achieving zero-phase filtering has been to combine several recursively computable IIR filters in either a cascade or parallel structure. In the cascade approach, a filter whose impulse response is $h[\mathbf{n}]$ is cascaded with one whose response is $h[-\mathbf{n}]$. The result is a system whose overall frequency response is a real, nonnegative function. Although this is not an unreasonable approach to take, one does encounter difficulties in the eventual implementation. In particular, an infinite number of samples of the first filter must be computed before filtering with the second filter can commence. Any truncation of the first filter's output invariably introduces some amount of discernible distortion in the filtered image. The more popular parallel approach consists of decomposing the zero-phase rational filter into 4 single quadrant filters and then combining them in a parallel structure [4, 5, 6]. This technique works well but is best suited for 2-D rational filters designed in the spatial domain.

We begin this paper by examining a simple yet effective scheme for designing 2-D zero-phase rational filters. The technique is based on the McClellan transformation of a 1-D zero-phase IIR filter and is similar to design algorithms proposed in [4, 6, 7]. While this design method is integral to the topic at hand, the main focus and contribution of the paper lies with the filter's eventual realization. The novelty of our approach is that the designed filter is realized, not recursively using initial conditions as is the case in both the cascade and parallel approaches, but rather simultaneously through the assumption of boundary conditions. The effect is to transform the cumbersome filtering task into a computationally tractable boundary value problem. We shall, however, defer further discussion of our realization procedure until after the design technique has been introduced.

2 Design Technique

The design procedure consists of the following steps:

i) Designing a causal and stable 1-D discrete-time IIR filter

$$H_c(z) = \frac{H_{cb}(z)}{H_{ca}(z)} \tag{1}$$

using any of a number of standard design techniques.

ii) Generating a 1-D zero-phase IIR filter by cascading $H_c(z)$ with its conjugate, i.e.

$$H_z(z) = H_c(z)H_c(z^{-1}) = \frac{|H_{cb}(z)|^2}{|H_{ca}(z)|^2}. \tag{2}$$

The frequency response $H_z(\omega)$ can be expressed as

$$H_z(\omega) = \frac{\sum_{k=0}^{M-1} b[k]cos(k\omega)}{\sum_{l=0}^{M-1} a[l]cos(l\omega)} = \frac{\sum_{k=0}^{M-1} b[k]T_k[cos(\omega)]}{\sum_{l=0}^{M-1} a[l]T_l[cos(\omega)]}, \tag{3}$$

where $T_k[x]$ is the kth Chebyshev polynomial.

iii) Applying a McClellan transformation to to the numerator and denominator polynomials of $H_z(\omega)$ by making the substitution $F(\omega_1, \omega_2) \rightarrow cos(\omega)$, i.e. let

$$H_z(\omega_1, \omega_2) = \frac{\sum_{k=0}^{M-1} b[k]T_k[F(\omega_1, \omega_2)]}{\sum_{l=0}^{M-1} a[l]T_l[F(\omega_1, \omega_2)]}. \tag{4}$$

Unlike similar methods, the zero-phase transformation function is allowed to be of the form

$$F(\omega_1, \omega_2) = \frac{F_b(\omega_1, \omega_2)}{F_a(\omega_1, \omega_2)}. \tag{5}$$

This algorithm is similar to the ones proposed in [8, 4, 6] with the exception that we allow the transformation function to be a rational polynomial. It can easily be shown that $H_z(\omega_1, \omega_2)$ will be a bounded frequency response provided that the zero-phase transformation function satisfies

$$-1 \leq F(\omega_1, \omega_2) \leq 1. \tag{6}$$

As an aside, we note that the zero-phase transformation function can be designed using a weighted least squares technique in the frequency domain. The only difficulty is in ensuring that $F(\omega_1, \omega_2)$ satisfies

$$0 \leq |F(\omega_1, \omega_2)| \leq F_{max} < \infty. \tag{7}$$

If $F(\omega_1, \omega_2)$ is chosen to have a separable denominator then we can check condition (7) by factoring $F_a(\omega_1, \omega_2) = F_{a1}(\omega_1)F_{a2}(\omega_2)$ and verifying that

$$F_a(\omega_1, \omega_2) \neq 0 \qquad -\pi \leq \omega_1, \omega_2 \leq \pi. \tag{8}$$

In the more general case, however, one must resort to *ad hoc* methods for verification. In our simulations, we have found that the Davidon-Fletcher-Powell algorithm provides a bounded $F(\omega_1, \omega_2)$ when the minimization is performed over a sufficiently large set of frequencies.

3 Boundary Valued Realization

As was implied earlier, zero-phase rational filters have been employed sparingly in image processing applications due to the difficulty revolving around actual implementation. In this section we introduce a solution to this problem which allows us to simultaneously compute a finite section of the filtered image.

To understand our approach, consider a 2-D zero-phase rational filter, $H(\omega_1, \omega_2)$, whose input, $x[\mathbf{n}]$, and output, $y[\mathbf{n}]$, are related by the finite order difference equation

$$\sum_{\mathbf{k}\in\mathbf{R_a}} a[\mathbf{k}]y[\mathbf{n}-\mathbf{k}] = \sum_{\mathbf{j}\in\mathbf{R_b}} b[\mathbf{j}]x[\mathbf{n}-\mathbf{j}]. \tag{9}$$

It should be apparent that (9) does not admit a recursive solution based upon initial conditions. On the other hand, we can find a signal $\hat{y}[\mathbf{n}]$ which satisfies (9) on a finite region of support, $\mathbf{R_y}$, provided that boundary conditions are specified.[1] The signal $\hat{y}[\mathbf{n}]$ will not in general be an exact truncation of the linear shift invariant filter output $y[\mathbf{n}]$, but will be some close approximation. The assumption is that the boundary-valued system output will be adequate for most practical applications. If we let $\mathbf{x}$ and $\hat{\mathbf{y}}$ be the lexicographically ordered input and output signals, then the proposed realization amounts to solving a system of linear equations of the form

$$\mathbf{A}\hat{\mathbf{y}} = \mathbf{B}\mathbf{x} + \mathbf{z}, \tag{10}$$

where $\mathbf{A}$ and $\mathbf{B}$ are sparse, "almost" block-Toeplitz matrices that are functions of the coefficients $a[\mathbf{k}]$, $b[\mathbf{j}]$, and the boundary conditions and where the entries in $\mathbf{z}$ are simply linear combinations of the boundary values.

It is seen that there are two distinct facets to this method of filter realization; the first being the specification of boundary values and the second, the actual solution of (10).

3.1 Specification of Boundary Values

The boundary values may be known a priori, assumed, or inferred. In most instances the incorporation of boundary values is simply an artifice that allows us to compute the signal $\hat{y}[\mathbf{n}]$ on $\mathbf{R_y}$. The only requirement is that the boundary values be chosen to be as unobtrusive as possible; the assumed values must not bias the solution nor should they give rise to any discernible spatial distortion near the boundaries. The obvious question is how to choose boundary values which satisfy the above constraint. The question, unfortunately, remains unanswered at present.

Upon experimenting with numerous types of boundary conditions, we have found the zero-value and the class of periodic boundary conditions to yield the best results. The zero-value boundary condition is an assumed condition; all boundary values are set to zero. The assumption leads to a highly structured block-Toeplitz system of linear equations which can be efficiently solved using Levinson-like recursions [9, 10]. Simulations revealed that use of the zero-value boundary condition sometimes led to ringing artifacts along the boundary of the filtered image. The ringing was localized to the boundaries, hence it was easily removed by a simple cropping operation.

---

[1] The region of support is typically taken to be an $N \times N$ square.

In contrast, the periodic condition is inferred from the assumption that the input signal $x[\mathbf{n}]$ satisfies some type of periodicity property, in most instances the toroidal periodicity. By assuming the input signal to have toroidal periodicity we can deduce that the output signal, $y[\mathbf{n}]$, also has toroidal periodicity. Hence, the matrices **A** and **B** become block-circulant and the system (10) can be solved using 2-D FFT's. Similarly, other assumed periodicities lead to systems of equations which can be efficiently solved using 2-D discrete sine and cosine transforms. We also note that the periodicity assumption can lead to wrap-around distortion along the edges of the filtered signal.

3.2 Realization

As indicated previously, it is possible to use so-called direct methods of solution to obtain the signal $\hat{\mathbf{y}}$. Unfortunately, the size of the signal to be filtered often precludes the use of direct solution techniques.

Instead one must resort to iterative solution techniques. While these methods are not guaranteed to give exact solutions after a finite number of computations, they are able to produce a sequence of solution vectors $\hat{\mathbf{y}}^{(\mathbf{k})}$ that converges to the desired solution $\hat{\mathbf{y}}$. There are a multitude of iterative algorithms which have been successfully used to numerically solve partial differential equations. Among these, the conjugate-gradient (CG) is the most popular [11, 12].

Due to space constraints, we omit details concerning the algorithm itself, concentrating instead on determining a bound on the number of iterations needed to reach the relative error defined as

$$\mathrm{RE} \stackrel{\mathrm{def}}{=} \frac{\|\hat{\mathbf{y}} - \hat{\mathbf{y}}^{(\mathbf{k})}\|}{\|\hat{\mathbf{y}}\|}, \tag{11}$$

to a predetermined amount ϵ. It can be shown that an upper bound on the number of iterations needed to reach a relative error ϵ is given by

$$k \approx \lceil \ln \frac{2}{\epsilon} / ln\sigma^{-1} \rceil \tag{12}$$

where

$$\sigma = (1 - \frac{1}{\sqrt{\chi}})/(1 + \frac{1}{\sqrt{\chi}}) \tag{13}$$

and χ is the condition number of the $N^2 \times N^2$ matrix **A** [12]. While it is difficult to obtain the condition number exactly we can closely approximate it by the quantity

$$\hat{\chi} = \frac{\max_\omega A(\omega)}{\min_\omega A(\omega)}. \tag{14}$$

Consider the design of a circularly symmetric low-pass filter having a cutoff frequency of ω_c. Suppose in our design, we use the magnitude-squared response of a 1-D digital Butterworth lowpass filter as our symmetric 1-D filter, i.e.

$$H_z(\omega) = \frac{(1+\cos\omega)^M}{(1+\cos\omega)^M + \beta^N(1-\cos\omega)^M} \tag{15}$$

where

$$\beta = \left(\frac{1+\cos\omega_c}{1-\cos\omega_c}\right), \tag{16}$$

and the original McClellan transformation,

$$F(\omega_1,\omega_2) = \frac{1}{2}(-1+\cos\omega_1+\cos\omega_2+\cos\omega_1\cos\omega_2) \tag{17}$$

For this design, it can be shown that

$$\hat{\chi} \approx \frac{\beta^N 2^{N-1}}{(1+\cos\omega_c)^{N-1}} \tag{18}$$

Using (18), the number of iterations needed by the (CG) algorithm to low-pass filter a given image can be estimated.

Further details concerning the realization of the filter are omitted from this summary due to space constraints.

4 Examples

In this section we include some illustrative examples.

Initially, we show the frequency response of two filters designed using the proposed technique. The first filter, shown in Figure 1, is a circularly symmetric low-pass filter having a 5th order 1-D digital Butterworth filter as its prototype and transformation function

$$F(\omega_1,\omega_2) = -\frac{1}{2}+\frac{1}{2}(\cos\omega_1+\cos\omega_2)+\frac{1}{4}(\cos(\omega_1+\omega_2)+\cos(\omega_1-\omega_2)). \tag{19}$$

The second filter, shown in Figure 2, is a symmetric fan filter having a 5th order 1-D digital Butterworth filter as its prototype and transformation function

$$F(\omega_1,\omega_2) = \frac{1}{2}(\cos\omega_1-\cos\omega_2). \tag{20}$$

The final example shows an image before and after it has been processed with the filter shown in Figure 1. For this example, toroidal boundary conditions have been imposed. With the exception of slight wrap-around effect along the edges of the filtered image, the incorporation of this boundary condition introduces no objectionable distortion.

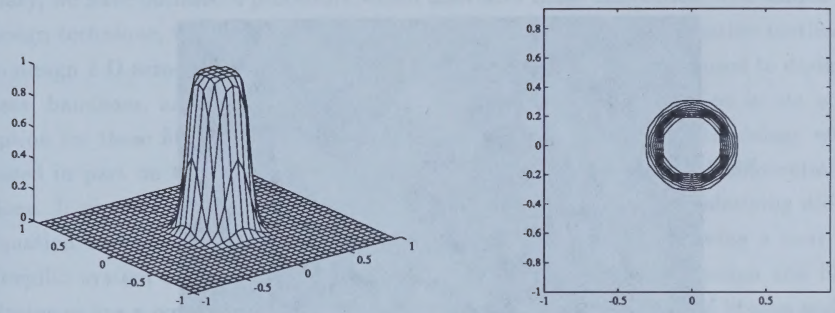


Figure 1: Frequency Response of a Circularly Symmetric Low-Pass Filter Designed Using Proposed Technique

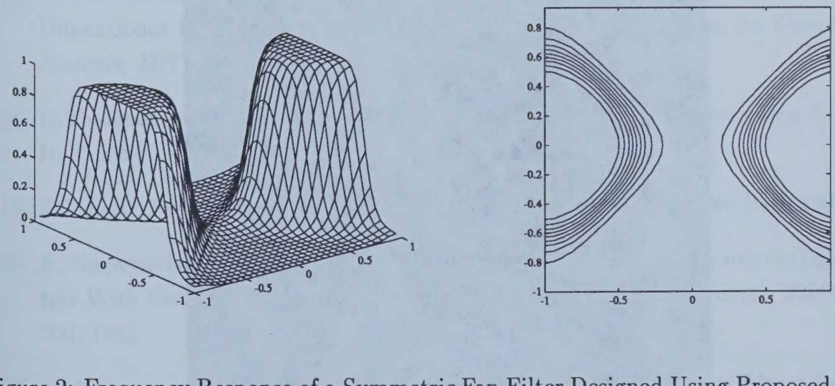


Figure 2: Frequency Response of a Symmetric Fan Filter Designed Using Proposed Technique

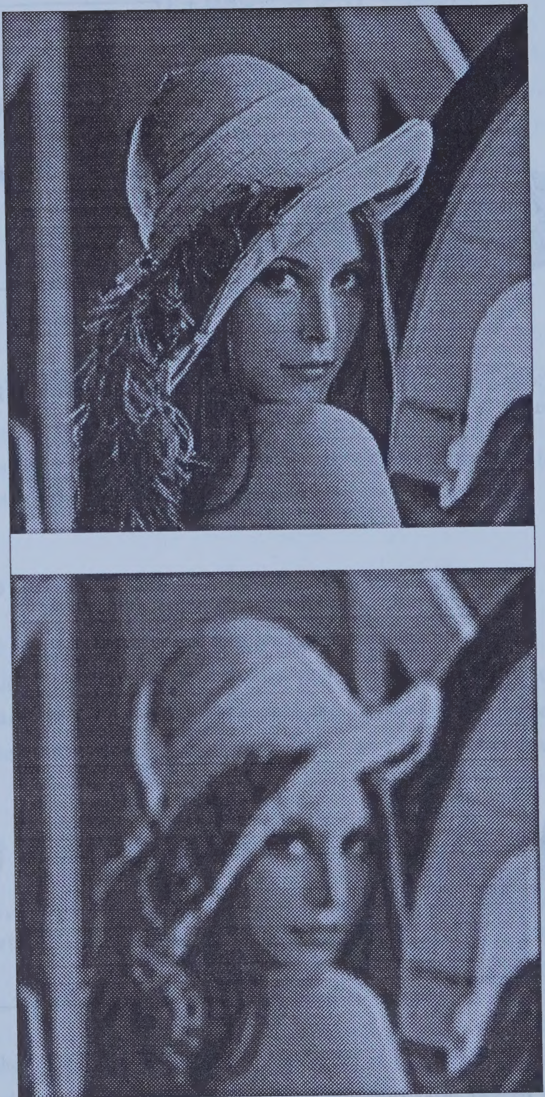


Figure 3: Test Image Before and After Zero-Phase Filtering

5 Conclusion

As mentioned in the introduction, the zero-phase rational filter has been a relatively unused image processing tool due to both design and realization difficulties. In this summary, we have outlined a procedure which alleviates these difficulties. The simple 3-step design technique, which is similar to the popular McClellan transformation method used to design 2-D zero-phase finite impulse response (FIR) filters, can be used to design lowpass, bandpass, and fan filters. Since initial condition based recursion is not a viable option for these filters, we have introduced an alternate filtering methodology which is based in part on techniques used to numerically solve elliptic partial differential equations. It consists of supplying boundary conditions along with the underlying difference equation and simultaneously realizing the filtering operation by solving a near block-Toeplitz system of linear equations. Our results indicate that the design and filtering strategies are a novel way of performing frequency selective filtering of images and other multidimensional signals.

References

[1] J. McClellan. The Design of Two-Dimensional Digital Filters by Transformations. In *Proc. 7th Ann. Princeton Conf. Inf. Sci. Sys.*, 1973.

[2] R.M. Mersereau and W.F.G. Mecklenbrauker. Mcclellan Transformations For Two-Dimensional Digital Filtering: I-Design. *IEEE Trans. on Circuits and Systems*, 23(7):405–414, July 1976.

[3] W.F.G. Mecklenbrauker and R.M. Mersereau. Mcclellan Transformations For Two-Dimensional Digital Filtering: II-Implementation. *IEEE Trans. on Circuits and Systems*, 23(7):414–422, July 1976.

[4] D. Dudgeon and R. Mersereau. *Multidimensional Digital Signal Processing.* Prentice-Hall, 1984.

[5] J.S. Lim. *Two-Dimensional Signal and Image Processing.* Prentice-Hall, 1990.

[6] N. Nagamuthu, M. Sid-Ahmed, and M. Ahmad. Design of 2-D Recursive Digital Filters With Constant Group Delay Characteristics. *J. Franklin Institute*, 320(4):191–200, 1985.

[7] D. Nguyen and M. Swamy. A Class of 2-D Seperable Denominator Filters Designed via Mcclellan Transform. *IEEE Trans. on Circuits and Systems*, 33(9):874–881, Sept. 1986.

[8] A.H. Kayran and R.A. King. Design of Recursive and Nonrecursive Fan Filters With Complex Transformations. *IEEE Trans. on Circuits and Systems*, 30(12):849–857, Dec 1983.

[9] R. Wiggins and E. Robinson. Recursive Solution To the Multichannel Filtering Problem. *J. Geophysics Res.*, 70(8):1885–1891, 1965.

[10] H. Akaike. Block Toeplitz Matrix Inversion. *SIAM J. Appl. Math*, 24(2):234–241, March 1973.

[11] L.A. Hageman and D.M. Young. *Applied Iterative Methods.* Academic Press, 1981.

[12] O. Axelsson and G. Lindskog. On the Rate of Convergence of the Preconditioned Conjugate Gradient Method. *Numerische Mathematik*, 48:499–523, 1986.

CONVERGENCE OF SUCCESSIVE APPROXIMATIONS FOR NONEXPANSIVE OPERATORS [†]

Lee C. Potter
Department of Electrical Engineering
The Ohio State University
Columbus, OH 43210

Abstract

Iterative algorithms are of wide interest in engineering computation. However, demonstration of convergence and characterization of fixed points can present difficulties in analysis. In this paper we survey properties of nonexpansive, nonlinear mappings that are relevant to establishing the existence of fixed points and the convergence of iterative schemes to compute them. Application of these properties is illustrated in an alternative treatment of the popular successive projections algorithm. In addition, an iterative scheme is presented for computing the projection onto a finite intersection of closed convex sets by a simple cyclic sequence of projections onto the individual sets.

Key Words

nonexpansive operator, successive approximations, nearest-point projection, fixed point

I. Introduction

A *fixed point* of an operator f is an element x in the domain of f satisfying $f(x) = x$. Iterative solution for a fixed point is the basis for computation in applications ranging from polynomial rooting [20] to image reconstruction [2, 18, 28, 30]. Likewise, a dual approach to constrained optimization problems often gives rise to fixed point problems [5, 14, 25, 27, 29]. Commonly employed results require that f is a contraction [28], is linear [2], or has compact domain [30]. However, convergence analysis often requires more general operators and may become problematic.

In this paper we survey several properties of nonexpansive mappings relevant to establishing the existence of fixed points and the convergence of iterative schemes to compute them. Averaged mappings, firmly nonexpansive maps, and nearest-point projections onto convex sets are considered as important special cases. The application of these properties in engineering computation is illustrated by an alternative formulation of the

---

[†]This work was supported in part by the National Science Foundation under Grant MIPS-9111044.

popular successive projection algorithm [34]. Additionally, an algorithm is presented for computing the projection onto the intersection of m closed convex sets, given only the m projection operators onto the constituent sets.

Perhaps the simplest and most useful result in fixed point theory is the Banach fixed point theorem, or contraction mapping principle, which asserts that if f is a self-mapping of a complete normed linear space $\mathcal{H}$ satisfying

$$\|f(x) - f(y)\| \leq L\|x - y\|, \quad L < 1 \quad \forall\, x, y \in \mathcal{H}$$

then f has a unique fixed point, and, moreover, for every $x \in \mathcal{H}$ the sequence of *successive approximations*, or *Picard iterates*, $\{f^n(x)\}$, converges to the fixed point. Within the less restrictive class of nonexpansive operators ($L \leq 1$), the most frequently cited results require boundedness. For example, the Brouwer fixed point theorem assures that for any continuous self-mapping of the unit sphere in $\mathbb{R}^n$ there exists a fixed point. First published in 1910, the result was previously proven by Poincaré (1886) in a similar form. In Hilbert spaces, an analogous result is given by Browder:

Property 1 ([3]) *If $\mathcal{K}$ is a bounded closed convex set in a Hilbert space $\mathcal{H}$ and $f : \mathcal{K} \mapsto \mathcal{K}$ is nonexpansive, then there exists some $x \in \mathcal{K}$ such that $f(x) = x$.*

II. Fixed Points of Nonexpansive Maps

Often in application the operator f is not a contraction, so the Banach fixed point theorem is not applicable. Likewise, the domain of f may be neither closed nor bounded, so the well-known Brouwer fixed point results for nonexpansive operators [17, 30] are not applicable. This section explores the asymptotic behavior of Picard iterates, $\{f^n(x)\}$, for a general nonexpansive operator f. Let $\mathcal{H}$ be a real Hilbert space[1] with inner product $\langle x, y\rangle$ and norm $\|x\| = \langle x, x\rangle^{1/2}$, $x, y \in \mathcal{H}$. Suppose $\mathcal{D} \subset \mathcal{H}$. An operator $f : \mathcal{D} \mapsto \mathcal{H}$ is said to be *nonexpansive* if

$$\|f(x) - f(y)\| \leq \|x - y\| \quad \forall\, x, y \in \mathcal{D}.$$

The class of nonexpansive mappings is more than just a trivial generalization of a Lipschitz condition; this class of operators is intimately connected with differential equations, monotone operators, and the geometry of Banach spaces. Nonexpansive mappings may be fixed point free, and when such a mapping has a fixed point it need not be unique (e.g., the identity mapping). The following Properties characterize the existence of fixed points and provide a necessary and sufficient condition for convergence of the Picard iterations. In the sequel, let $\mathcal{K}$ denote a nonempty, closed and convex subset of $\mathcal{H}$.

Property 2 ([11, 12]) *For a nonexpansive mapping $f : \mathcal{K} \mapsto \mathcal{K}$, the following are equivalent:*

---

[1] Results are reported here for Hilbert spaces; most can be directly applied in any uniformly convex Banach space.

(i) f has a fixed point.

(ii) $\{\|f^n(x)\|\}$ is a bounded sequence for some x in $\mathcal{K}$.

(iii) There is a bounded sequence $\{y_n\}$ in $\mathcal{K}$ such that $\lim_n(y_n - f(y_n)) = 0$.

(iv) the origin is in the range of $(I - f)$, where I is the identity map.

Property 3 ([12]) *For a nonexpansive mapping $f : \mathcal{K} \mapsto \mathcal{K}$, the set of fixed points of a nonexpansive operator is closed and convex.*

Property 4 ([12]) *Let $f : \mathcal{K} \mapsto \mathcal{K}$ be a nonexpansive operator with a fixed point, and let x be a point in $\mathcal{K}$. Then, $\{f^n(x)\}$ converges weakly to a fixed point of f if and only if f is* weakly asymptotically regular[2] *at x, i.e.,*

$$\text{weak} \lim_n (f^n(x) - f^{n+1}(x)) = 0.$$

An operator f that is (weakly) asymptotically regular for all $x \in \mathcal{K}$ is said to be (weakly) asymptotically regular. As an example of a nonexpansive operator $\mathbb{R}$ to $\mathbb{R}$ with a fixed point and not asymptotically regular, consider $f(x) = -(x + 1)$.

For nonexpansive mappings without a fixed point, the Picard iterations still yield useful information. In particular, we have the following positive result.

Property 5 ([24]) *Let $f : \mathcal{K} \mapsto \mathcal{K}$ be a nonexpansive mapping. Then $\lim_n \frac{1}{n} f^n(x) = -v$, where v is the unique element of minimum norm in the closure of the range of $(I - f)$.*

A. Averaged maps

Although a nonexpansive f with a fixed point may not be asymptotically regular, the averaged mapping $f_\lambda = \lambda f + (1 - \lambda)I$, where $0 < \lambda < 1$, shares the same fixed point set, and the successive approximations $\{f_\lambda^n(x)\}$ are guaranteed to converge to a fixed point from any initial iterate.

Property 6 *Let f be a nonexpansive mapping on $\mathcal{H}$, and define $f_\lambda = \lambda f + (1 - \lambda)I$, where $\lambda \in (0, 1)$. Then,*

(i) f_λ has the same fixed point set as f.

(ii) f_λ is nonexpansive for all $\lambda \in (0, 1)$.

Property 7 ([21]) *Let $f : \mathcal{K} \mapsto \mathcal{K}$ be a nonexpansive operator with at least one fixed point. Although f itself may not be weakly asymptotically regular, the averaged mapping f_λ, $\lambda \in (0, 1)$, is weakly asymptotically regular, and $\{f_\lambda^n(x)\}$ is therefore weakly convergent to a fixed point of f for every $x \in \mathcal{K}$.*

The asymptotic behavior of the averaged mapping f_λ also provides insight into the range of f, even when f has no fixed point.

---

[2] A sequence $\{x_n\} \in \mathcal{H}$ is said to converge weakly to x if $\lim_{n\to\infty}\langle x_n, y\rangle = \langle x, y\rangle$ for every $y \in \mathcal{H}$. Weak convergence implies strong convergence in finite dimensional spaces.

Property 8 ([1]) *Let $f : \mathcal{K} \mapsto \mathcal{K}$ be nonexpansive and define the averaged mapping $f_\lambda = \lambda f + (1-\lambda)I$, $\lambda \in (0,1)$. Let v be the unique point of least norm in the closure of the range of $(I - f)$. Then for all x in $\mathcal{K}$*

(i) $\lim_{n\to\infty} \frac{1}{n} f_\lambda^n(x) = -v$.

(ii) $\lim_{n\to\infty}[f_\lambda^n(x) - f_\lambda^{n+1}(x)] = v$.

(iii) f has no fixed point if and only if $\lim_{n\to\infty} \|f_\lambda^n(x)\| = \infty$ *for all x in $\mathcal{K}$.*

In relation to the asymptotic regularity condition of Property 4, observe that $\{f_\lambda^n(x)\}$ is a Cauchy sequence if and only if v is the zero vector and f has a fixed point. For successive approximations, the averaged mapping f_λ is often used in place of f for an enhanced rate of convergence; this technique is known as *overrelaxation*.

B. Firmly nonexpansive maps

Next we consider the special subclass of firmly nonexpansive mappings.

Property 9 ([11, 12]) *Let $\mathcal{D} \subset \mathcal{H}$. For a mapping $f : \mathcal{D} \mapsto \mathcal{H}$ the following are equivalent.*

(i) f is firmly nonexpansive[3].

(ii) $2f - I$ is nonexpansive.

(iii) $f = \frac{1}{2}(g + I)$ for some nonexpansive g.

(iv) $\|f(x) - f(y)\|^2 \leq \langle f(x) - f(y),\, x - y\rangle \ \forall\, x, y \in \mathcal{K}$.

Property 10 *A composition of firmly nonexpansive mappings is likewise firmly nonexpansive.*

Moreover, the averaged mapping $\lambda f + (1-\lambda)I$ is firmly nonexpansive for nonexpansive f and $0 \leq \lambda \leq \frac{1}{2}$ [4].

In general, the iterates $\{f^n(x)\}$ of a nonexpansive mapping with a fixed point do not converge. However, as cited in the next Property, for firmly nonexpansive mappings having a fixed point the Picard iterates are guaranteed to converge to a fixed point.

Property 11 ([12]) *Let $f : \mathcal{K} \mapsto \mathcal{K}$ be a firmly nonexpansive mapping with a fixed point. Then, for every $x \in \mathcal{K}$, the sequence of iterates $\{f^n(x)\}$ converges weakly to a fixed point of f.*

Next, the asymptotic behavior is considered for the more general case in which f may be fixed point free. The following two Properties characterize when the sequence of Picard iterates $\{f_\lambda^n(x)\}$ is *unbounded* for all initial iterates x and guarantee that the successive difference in iterates converges.

---

[3] In a Banach space, a single-valued operator f mapping is said to be *firmly nonexpansive* if $\|f(x) - f(y)\| \leq \|\lambda(x-y) + (1-\lambda)(f(x) - f(y))\|$ for all x, y in the domain of f and all $\lambda > 0$.

Property 12 ([12]) *Let f be a firmly nonexpansive mapping of a closed convex subset $\mathcal{K}$ of a Hilbert space. Then f is fixed point free if and only if* $\lim_{n\to\infty} \|f^n(x)\| = \infty$ *for all* $x \in \mathcal{K}$.

Necessity is lost if f is merely nonexpansive: consider $f(x) = |x-2| - 1$, $x \in \mathbb{R}$.

Property 13 ([12]) *Let $\mathcal{K}$ be a closed convex subset of a Hilbert space $\mathcal{H}$ and $f : \mathcal{K} \mapsto \mathcal{K}$ a firmly nonexpansive mapping. Then for all $x \in \mathcal{K}$,* $\lim_{n\to\infty}[f^n(x) - f^{n+1}(x)] = v$ *where v is the unique point of least norm in the closure of the range of $(I - f)$. In addition,* $\lim \frac{1}{n} f^n(x) = -v$.

C. Projections onto convex sets

In Hilbert space, nearest-point projection operators onto convex sets are an important subclass of firmly nonexpansive operators.

Property 14 *Let $\mathcal{K}$ denote any nonempty, closed, convex subset of a Hilbert space $\mathcal{H}$. Then there exists a unique $y \in \mathcal{K}$ such that* $\inf_{z\in\mathcal{K}} \|x - z\| = \|x - y\|$.

This correspondence is denoted by $y = P_{\mathcal{K}}(x)$, where $P_{\mathcal{K}} : \mathcal{H} \mapsto \mathcal{K}$ is said to be the *nearest-point projection operator*, or simply the *projection*, of $\mathcal{H}$ onto the closed convex set $\mathcal{K}$. The projection is uniquely characterized by the inequality

$$\langle x - P_{\mathcal{K}}(x), y - P_{\mathcal{K}}(x)\rangle \leq 0 \; \forall\, y \in \mathcal{K}.$$

The operator $P_{\mathcal{K}}$ is linear if and only if $\mathcal{K}$ is a subspace.

Property 15 ([12]) *Let $P_{\mathcal{K}}$ be the projection onto a closed convex set $\mathcal{K} \subset \mathcal{H}$. Then, P is firmly nonexpansive, $(I - P_{\mathcal{K}})$ is nonexpansive, and the reflection of $\mathcal{H}$ in $\mathcal{K}$ defined by $2P_{\mathcal{K}} - I$ is nonexpansive.*

III. Application: Successive Projections

The method of successive projections (SP) onto convex sets [34] provides a convergent iterative algorithm to compute a signal in the intersection of convex constraint sets by projecting successive estimates of the signal onto each convex set in turn. The behavior of this technique is easily described in the context of firmly nonexpansive operators established in Section 2. The conceptual basis for SP is the characterization of each known property of a signal as a closed convex subset of a Hilbert space of signals. Thus, m such properties restrict the signal, x, to lie in the intersection $\mathcal{K} = \bigcap_{j=1}^{m} \mathcal{K}_j$. A signal common to all m convex sets is then determined iteratively by the method of successive approximation. Defining f as the composition of the projections P_j onto the respective sets $\mathcal{K}_j$,

$$f = P_m \circ P_{m-1} \circ \cdots \circ P_1, \tag{1}$$

the fixed point set of f is precisely $\mathcal{K}$, the intersection of the m constraint sets, *provided* the intersection $\mathcal{K}$ is nonempty [4, 34]. Hence, the determination of a signal in $\mathcal{K}$ is reduced to the determination of a fixed point of f, and the previous Properties directly provide an iterative scheme.

Theorem 1 *Let* $f : \mathcal{H} \mapsto \mathcal{H}$ *be the composition* $f = P_m \circ P_{m-1} \circ \cdots \circ P_1$, *and assume the set* $\mathcal{K} = \bigcap_{j=1}^{m} \mathcal{K}_j$ *is nonempty. For every initial iterate* $x \in \mathcal{H}$, *the sequence of iterates* $\{f^n(x)\}$ *converges weakly to some* $\hat{x} \in \mathcal{K}$.

Proof. The theorem follows trivially from the Properties enumerated above. By Properties 15 and 10, f is firmly nonexpansive. Further, y is a fixed point of f if and only if $y \in \mathcal{K}$. Then, for $\mathcal{K}$ nonempty, weak convergence follows from Property 11. ∎

Since weakly convergent sequences are bounded, it then follows that strong convergence is assured if the iterates are eventually contained in some compact or finite-dimensional subset of $\mathcal{H}$. Moreover, convergence is strong if and only if at least one subsequence of $f^n(x)$ converges strongly. The following corollary gives the over-relaxed version of the iterations in Equation (1).

Corollary 1 *Let* $f_\lambda : \mathcal{H} \mapsto \mathcal{H}$ *be the composition*

$$f = [\lambda_m P_m + (1 - \lambda_m)I] \circ [\lambda_{m-1} P_{m-1} + (1 - \lambda_{m-1})I] \circ \cdots \circ [\lambda_1 P_1 + (1 - \lambda_1)I],$$

where $0 < \lambda_j < 2$ *and assume the intersection,* $\mathcal{K}$, *is nonempty. For every initial iterate* $x \in \mathcal{H}$, *the sequence of iterates* $\{f_\lambda^n(x)\}$ *converges weakly to some* $\hat{x} \in \mathcal{K}$.

Proof. By Property 15, $2P - I$ is nonexpansive. Then, by Property 7 the averaged mapping $\delta_j(2P_j - I) + (1 - \delta_j)I = 2\delta_j P_j + (1 - 2\delta_j)I$ is weakly asymptotically regular for $\delta_j \in (0, 1)$. Let $\lambda_j = \frac{1}{2}\delta_j$ to obtain the result. ∎

Convergence rates and finite iteration error bounds are considered in [19, 22, 34]. The difficulties in generalizing to nonconvex constraint sets are explored in [6, 32].

A. Modified successive projections

In general, the limit point $\hat{x}$ depends on both the initial iterate x and the ordering of the constituent projections P_j; $\hat{x}$ is not the projection of x onto $\mathcal{K}$. While $\lim_n (P_m \circ \cdots \circ P_1)^n$ is indeed a nonexpansive mapping onto a nonempty $\mathcal{K}$, it is not, in general, the nearest-point projection onto $\mathcal{K}$. (Consider the simple counterexample provided by a disk and a line in the plane.) Yet, a surprisingly simple modification of the SP algorithm of Theorem 1 provides the projection onto a finite intersection of closed convex sets using only a cyclic sequence of projections onto each individual set.

Given a point $d \in \mathcal{H}$, closed convex sets $\mathcal{K}_1, \cdots, \mathcal{K}_m$, and the corresponding nearest point projections $P_1, \cdots, P_m$, initialize

$$y_1^{(0)} = \cdots = y_m^{(0)} = 0 \text{ and } x_m^{(0)} = d.$$

In each iteration compute $2m$ vectors

$$x_1^{(k)},\, y_1^{(k)},\, \cdots,\, x_m^{(k)},\, y_m^{(k)}$$

as follows: set $x_0^k = x_m^{k-1}$ and, for $i = 1, \cdots, m$ calculate

$$\begin{aligned} z &= x_{i-1}^{(k)} + y_i^{(k-1)}, \\ x_i^{(k)} &= P_i(z), \\ y_i^{(k)} &= z - P_i(z). \end{aligned} \tag{2}$$

The modified successive projections (MSP) algorithm in Equation (2) alters the successive projections algorithm of Theorem 1 at each iteration by adding the previous outer normal $y_i^{(k-1)}$ to $x_{i-1}^{(k)}$ before projection onto set $\mathcal{K}_i$. This adjustment forces the z vector into the normal cone of $\bigcap_{i=1}^m \mathcal{K}_i$ at the solution point $x^\star$. The MSP algorithm apparently originates in [8] for the special case of convex cones; for arbitrary closed convex sets in $\mathbb{R}^n$ the algorithm was independently presented in [16]. A streamlined proof of convergence for infinite-dimensional Hilbert spaces is given in [9].

Theorem 2 ([9]) *Let $\mathcal{K}_1, \cdots, \mathcal{K}_m$ be closed convex subsets of $\mathcal{H}$ with nonempty intersection $\mathcal{K} = \bigcap_{j=1}^m \mathcal{K}_j$, and let $d \in \mathcal{H}$. Then for each i the sequence $\{x_i^{(k)}\}$ in (2) converges (strongly) as $k \longrightarrow \infty$ to $x^\star$, the unique vector minimizing $\|x - d\|$ subject to $x \in \mathcal{K}$.*

If some $\mathcal{K}_j$ is a closed linear variety, then the outer normal $y_j^{(k)}$ need not be computed since

$$x_j^{(k)} = P_j(x_{j-1}^{(k)} + y_j^{(k-1)}) = P_j(x_{j-1}^{(k)}).$$

Thus, if every $\mathcal{K}_j$ is a linear variety, MSP reduces to SP. The result for this special case is due to Halperin [15] and dates to Von Neumann [31] for $m = 2$. As a corollary, the well-known band-limited extrapolation algorithm of Papoulis and Gerchberg [10, 23] converges to the minimum norm element of $\mathcal{K}$ for $d = 0$ since the constraint sets are linear varieties.

Although the m sequences $\{x_i^{(k)}\}$ are convergent, the m sequences of outer normals $\{y_i^{(k)}\}$ may not be bounded. The boundedness of $\{y_i^{(k)}\}$ is certainly necessary for practical computation and is guaranteed by the following regularity condition.

Corollary 2 ([9]) *If there is a $j \in \{1, \cdots, m\}$ such that*

$$\mathcal{K}_j \cap \bigcap_{i \neq j} int(\mathcal{K}_i) \neq \emptyset$$

then

$$\sup_{k,i} \|y_i^{(k)}\| \leq \infty$$

where $int(\mathcal{K}_i)$ denotes the interior of $\mathcal{K}_i$ with respect to the norm topology.

The convergence proof for Theorem 2 may be obtained by considering the Fenchel dual. The dual problem requires the minimization of a functional of m vectors, which is performed by cyclically minimizing with respect to a single vector while holding others constant; this classical approach yields the iterative algorithm in Equation (2).

B. Example projection operators

The MSP algorithm is applicable in numerous computation tasks including medical imaging, filter design, recovery of Euclidean distance matrices, and maximum likelihood estimation. One common framework is the convex quadratic optimization problem in $\mathbb{R}^n$

$$\min_x x'Qx + q'x \text{ subject to } x \in \mathcal{K} = \bigcap_{i=1}^{m} \mathcal{K}_i \tag{3}$$

where Q is a positive definite $n \times n$ matrix, $q \in \mathbb{R}^n$, and q' denotes the transpose of q. If we choose $\mathcal{H} = \mathbb{R}^n$ with inner product $\langle x, y\rangle = x'Qy$, then (3) is equivalent to

$$\min_{x\in\mathcal{K}} \|d - x\|$$

where $d = -\frac{1}{2}Q^{-1}q$. The computational cost of the algorithm depends on the complexity of each projection operator. Below we review [9, 16] three examples of sets $\mathcal{K}_i$ with simple projections under the inner product $\langle x, y\rangle = x'Qy$. First, for the halfspace $\mathcal{K}_i = \{x \in \mathbb{R}^n : a_i'x \leq b_i\}$ the projection is given by

$$P_i(x) = x - (a_i'Q^{-1}a_i)^{-1}(a_i'x - b_i)_+ Q^{-1}a_i$$

where $(t)_+ = \max(t, 0)$ for $t \in \mathbb{R}$. Second, for a linear constraint with both upper and lower bounds,

$$\mathcal{K}_i = \{x \in \mathbb{R}^n : c_i \leq a_i'x \leq b_i\},$$

the projection is given by

$$P_i(x) = x + (a_i'Q^{-1}a_i)^{-1}\{(c_i - a_i'x)_+ - (a_i'x - b_i)_+\}Q^{-1}a_i.$$

Such a constraint would be valuable, for example, in incorporating the physical bounds encountered in an adaptive array system [7]. Third, we review an ellipsoidal constraint

$$\mathcal{K}_i = \{x \in \mathbb{R}^n : (x - \bar{x}_i)'B_i(x - \bar{x}_i) \leq 1\}$$

where B_i is positive definite and $\bar{x}_i \in \mathbb{R}^n$. Let $B_i = L_iL_i'$ and find the spectral decomposition

$$L_i^{-1}Q(L^{-1})' = U_i'D_iU_i$$

where U_i is an orthogonal matrix and $D_i = \text{diag}(d_1, \cdots, d_k)$. Define the functional

$$g(x, \gamma) = \sum_{j=1}^{n}\{(U_iL_i'(x - \bar{x}_i))_j d_j(d_j + \gamma)^{-1}\}^2, \quad \gamma \geq 0$$

where $(\cdot)_j$ denotes the j^{th} component. Define $\gamma(x) = 0$ if $x \in \mathcal{K}$ and otherwise let $\gamma_i(x)$ denote the unique solution $\gamma \geq 0$ of $g_i(x, \gamma) = 1$. Then

$$P_i(x) = (L^{-1})'U_i'(D_i + \gamma_i(x)I)^{-1}D_iQ_iL_i'(x - \bar{x}_i) + \bar{x}_i.$$

C. Empty intersection

The convergence results in Theorems 1 and 2 require that the constraint sets have nonempty intersection; for $\mathcal{K}$ empty, we consider the important case of $m = 2$ convex sets. The following Property relates any fixed points of $P_1 \circ P_2$ to the distance between the sets $\mathcal{K}_1$ and $\mathcal{K}_2$.

Property 16 ([13]) *In a Hilbert space $\mathcal{H}$ let $\mathcal{K}_1$ and $\mathcal{K}_2$ be closed convex sets with projection operators P_1 and P_2, respectively. Then, $\hat{x}$ is a fixed point of $P_1 \circ P_2$ if and only if $\hat{x} \in \mathcal{K}_1$ and*

$$\|\hat{x} - P_2(\hat{x})\| = \inf_{\substack{x \in \mathcal{K}_1 \\ y \in \mathcal{K}_2}} \|x - y\|.$$

However, in general there may not exist a fixed point of $P_1 \circ P_2$; yet, boundedness of $\mathcal{K}_1$ suffices in Hilbert space. Thus, for bounded $\mathcal{K}_1$, the method of alternating projections yields a weakly convergent sequence.

Theorem 3 *Let $\mathcal{H}$ be a Hilbert space and suppose $\mathcal{K}_1$ and $\mathcal{K}_2$ are closed convex sets with $\mathcal{K}_1$ bounded. Let P_1 and P_2 denote the projection operators onto $\mathcal{K}_1$ and $\mathcal{K}_2$, respectively. Then for all $x \in \mathcal{H}$, $\{(P_1 \circ P_2)^n(x)\}$ is weakly convergent to a fixed point $\hat{x}$ of the composite mapping $P_1 \circ P_2$. Moreover, the minimum distance between $\mathcal{K}_1$ and $\mathcal{K}_2$ is achieved by $\|\hat{x} - P_2(\hat{x})\|$.*

Strong convergence of $\{P_1 \circ P_2(x)\}$ to a fixed point of $P_1 \circ P_2$ is assured when either (i) $\mathcal{K}_1$ is compact or (ii) $\mathcal{K}_1$ is finite dimensional and the distance between $\mathcal{K}_1$ and $\mathcal{K}_2$ is attained [13]. Moreover, for $\mathcal{K}_1 = \{x : Ax = b\}$ the measurement functionals comprising A may be orthonormalized so that the iterations yield a *least-squares* solution to $Ax = b$ [26]. For the case of $m > 2$ constraint sets (with at least one set bounded), a closed chain of nearest-neighbor elements is formed from the fixed points for the $m!$ compositions of the m projection operators [33].

References

[1] J. B. Baillon, R. E. Bruck, and S. Reich, *On the asymptotic behavior of nonexpansive mappings and semigroups in Banach spaces*, Houston J. Math., 4 (1978), pp. 1–9.

[2] J. Biemond, R. L. Lagendijk, and R. M. Mersereau, *Iterative methods for image deblurring*, Proceedings of the IEEE, 78 (1990), pp. 856–883.

[3] F. E. Browder, *Fixed-point theorems for noncompact mappings in Hilbert space*, Proc. Nat. Acad. Sci., 53 (1965), pp. 1272–1276.

[4] R. E. Bruck and S. Reich, *Nonexpansive projections and resolvents of accretive operators*, Houston J. Math., 3 (1977), pp. 459–470.

[5] C. L. Byrne and R. M. Fitzgerald, *Reconstruction from partial information with applications to tomography*, SIAM J. Appl. Math., 42 (1982), pp. 933–940.

[6] P. L. Combettes and H. J. Trussel, *Method of successive projections for finding a common point of sets in metric spaces*, J. Optimization Theory and Applications, 67 (1990), pp. 487–507.

[7] R. T. Compton, *Adaptive Antennas: Concepts and Performance*, Prentice Hall, Englewood Cliffs, NJ, 1988.

[8] R. L. DYKSTRA, *An algorithm for restricted least squares regression*, Journal of the American Statistical Association, 78 (1983), pp. 837–842.

[9] N. GAFFKE AND R. MATHAR, *A cyclic projection algorithm via duality*, Metrika, 36 (1989), pp. 29–54.

[10] R. W. GERCHBERG, *Super resolution through error energy reduction*, Optica Acta, 21 (1975), pp. 709–720.

[11] K. GOEBEL AND W. KIRK, *Topics in Metric Fixed Point Theory*, Cambridge University Press, Cambridge, 1990.

[12] K. GOEBEL AND S. REICH, *Uniform Convexity, Hyperbolic Geometry, and Nonexpansive Mappings*, Marcel Dekker, New York, 1984.

[13] A. A. GOLDSTEIN, *Constructive Real Analysis*, Harper and Row, New York, 1967.

[14] R. K. GOODRICH AND A. STEINHARDT, *L_2 spectral estimation*, SIAM J. Appl. Math., 46 (1986), pp. 417–426.

[15] I. HALPERIN, *The product of projection operators*, Acta Scientiarum Mathematicarum (Szeged), 23 (1962), pp. 96–99.

[16] S.-P. HAN, *A successive projection method*, Mathematical Programming, 40 (1988), pp. 1–14.

[17] V. I. ISTRĂŢESCU, *Fixed Point Theory*, Reidel, Dordrecht, 1981.

[18] S. KAWATA AND O. NALCIOGLU, *Constrained iterative reconstruction by the conjugate gradient method*, IEEE Trans. Med. Imaging, MI-4 (1985), pp. 65–71.

[19] S. KAYALAR AND H. L. WEINERT, *Error bounds for the method of alternating projections*, Mathematics of Control, Signals, and Systems, 1 (1988), pp. 43–59.

[20] D. G. LUENBERGER, *Optimization by Vector Space Methods*, Wiley, New York, 1969.

[21] Z. OPIAL, *Weak convergence of the sequence of successive approximations for nonexpansive mappings*, Bull. Amer. Math. Soc., 73 (1967), pp. 591–597.

[22] N. OTTAVY, *Strong convergence of projection-like methods in Hilbert spaces*, Journal of Optimization Theory and Applications, 56 (1988), pp. 433–461.

[23] A. PAPOULIS, *A new algorithm in spectral analysis and band-limited extrapolation*, IEEE Trans. Circuits Syst., CAS-22 (1975), pp. 735–742.

[24] A. PAZY, *Asymptotic behavior of contractions in Hilbert space*, Israel J. Math., 9 (1971), pp. 235–240.

[25] L. C. POTTER AND K. S. ARUN, *A dual approach to linear inverse problems with convex constraints*, SIAM J. Contr. Optimization. Under review.

[26] ——, *An iterative, least-squares solution to constrained inverse problems: Theory and application*, IEEE Transactions on Signal Processing. Under review.

[27] J. L. C. SANZ AND T. S. HUANG, *A unified approach to noniterative linear signal restoration*, IEEE Trans. Acoust., Speech, Signal Processing, ASSP-32 (1984), pp. 403–409.

[28] R. W. SCHAFER, R. M. MERSEREAU, AND M. A. RICHARDS, *Constrained iterative restoration algorithms*, Proc. IEEE, 69 (1981), pp. 432–450.

[29] A. O. STEINHARDT, R. K. GOODRICH, AND R. A. ROBERTS, *Spectral estimation via minimum energy correlation extension*, IEEE Trans. Acoust., Speech, Signal Processing, ASSP-33 (1985), pp. 1509–1515.

[30] V. T. TOM, T. F. QUATIERI, M. H. HAYES, AND J. H. MCCLELLAN, *Convergence of iterative nonexpansive signal reconstruction algorithms*, IEEE Trans. Acoust., Speech, Signal Processing, ASSP-29 (1981), pp. 1052–1058.

[31] J. VON NEUMANN, *On rings of operators. Reduction theory*, Annals of Mathematics, 50 (1949), pp. 401–485.

[32] C. A. WEBER AND J. P. ALLEBACH, *Reconstruction of frequency-offset fourier data by alternating projections*, in Proc. Twenty-fourth Allerton Conf. Commun. Control Computing, 1986, pp. 194–203.

[33] D. C. YOULA AND V. VELASCO, *Extensions of a result on the synthesis of signals in the presence of inconsistent constraints*, IEEE Trans. Circuits Syst., CAS-33 (1986), pp. 465–468.

[34] D. C. YOULA AND H. WEBB, *Image restoration by the method of convex projections: Part 1—Theory*, IEEE Trans. Med. Imaging, MI-1 (1982), pp. 81–94.

Fast Inversion of Hermitian Toeplitz and Quasi-Toeplitz Matrices with Arbitrary Rank Profile.[1]

Debajyoti Pal and Thomas Kailath[2]

AT&T Bell Laboratories
200 Laurel Avenue
Middletown, NJ 07747

Abstract

We present a fast procedure for computing inverse of nonsingular Hermitian Toeplitz and Quasi-Toeplitz (matrices congruent in a certain sense to Toeplitz matrices) matrices with an arbitrary rank profile. Our algorithm is obtained by simultaneously extending a certain procedure of Chun for inverting positive definite Toeplitz matrices and a generating function approach of Lev-Ari and Kailath for matrices with a generalized displacement structure. Unlike all other $O(n^2)$ algorithms for inversion this method works as long as the underlying matrix is nonsingular and does not require any other assumption.

1 Introduction

For a general $n \times n$ matrix, inversion requires $O(n^3)$ elementary operations. The special structure of Toeplitz matrices,

$$T_{n-1} = [t_{i-j}] \ , \ \ 0 \le i,j \le n \ ,$$

allows one to invert them, find a triangular factorization or solve a Toeplitz system of linear equations by fast algorithms that require only $O(n^2)$ operations (see e.g. [1], [2] or [3] and the references therein). A Hermitian Quasi-Toeplitz matrix is one of the form $L_t T L_t^*$ [3] where L_t and T are lower triangular Toeplitz and Hermitian Toeplitz respectively. Quasi-Toeplitz structure arises naturally in the process of factoring a Toeplitz matrix; in fact the Schur complement of the top left corner entry of a Toeplitz matrix is Quasi-Toeplitz. Thus it is natural to consider the problem of inverting Quasi-Toeplitz matrices along with Toeplitz matrices. Quasi-Toeplitz matrices also arise in other contexts, e.g. in the problem of root distribution of polynomials with respect to (w.r.t.) the unit circle

---

[1] This work was supported in part by the U. S. Army Research Office under Contract DAAL03-86-K-0045 and the SDIO/IST, managed by the Army Research Office under Contract DAAL03-87-K-0033.

[2] Thomas Kailath is with the Information Systems Laboratory, Stanford University, Stanford, CA 94305

[3] * denotes Hermitian transpose for matrices and complex conjugate for scalars.

(see [4] and [5]). See [6], [3], [7], [1], [2] and the references therein for a modest list references on inversion and factorization of Quasi-Toeplitz matrices.

However all the above authors assumed that the matrices involved were strongly regular, i.e. all its principal minors are nonzero. Only recently have Heinig and Rost [1], Gover and Barnett [8], and Labahn, Choi and Cabay [9] considered the problem of inverting Toeplitz matrices with singular sections. Also the problem of solving Toeplitz systems of linear equations involving matrices with arbitrary rank profile has been considered by Rissanen [10] and more recently by Delsarte *et al* [11], Sugiyama [12] and Pombra *et al* [13].

Gover and Barnett [8] resort to the so called r-Toeplitz (see [8] for details) operations or equivalently to block Toeplitz operations. These block operations are not fast (i.e. they require $O(r^3)$ computations for a block of size r). Moreover it requires considerable amount of computation just to find the appropriate block size. Apparently the first fast algorithm for inverting arbitrary Toeplitz matrices is due to Heinig and Rost [1]. They generalize the classical Levinson algorithm and their procedure continues to be a fast procedure in the presence of consecutive zero principal minors. Although finding the number of consecutive zero principal minors requires a certain amount of computation, the final computational complexity is $O(n^2)$. However due to need for extensive inner product computation, this algorithm does not parallelize well (i.e. each recursive step requires a constant number of parallel steps so that the whole algorithm takes $O(n)$ parallel steps) and requires $O(nlogn)$ parallel steps. Delsarte *et al* [11] independently derived essentially the same algorithm as Heinig and Rost for the special case of Hermitian Toeplitz matrices. Pombra *et al* [13] generalized the conventional three-term recursion for polynomials orthogonal on the unit circle: the scalar coefficients are replaced by certain polynomial coefficients; It is an inner product free procedure and thus it parallelizes well (as explained before). Also the determination of the number of consecutive zero minors can be accomplished by counting the number of repeated roots at the origin of a certain polynomial. Rissanen [10] and Sugiyama [12] gets to the solution by solving an associated Hankel system of equations. Labahn, Choi and Cabay [9] too present a solution for an associated Hankel matrix. However the last three approaches are not well suited for an order recursive scenario. The approaches of Delsarte *et al* and Pombra *et al* cannot be readily used for Quasi-Toeplitz matrices, since their derivations employ the Toeplitz structure rather than the underlying displacement structure. Heinig and Rost [1] pay attention to the underlying displacement structure, but their consideration of the Quasi-Toeplitz case is limited to strongly regular matrices only.

In this paper, we shall give a general solution to the problem of fast and completely recursive inversion of Toeplitz and Quasi-Toeplitz matrices with arbitrary rank profile. This is accomplished by generalizing, Chun's [3] Schur complement based procedure for inverting positive definite Toeplitz matrices to handle arbitrary nonsingular Toeplitz and Quasi-Toeplitz matrices.

In Sec.3 we derive a fast recursive procedure for computing Schur complements of Toeplitz and Quasi-Toeplitz (QT) matrices. Following Lev-Ari and Kailath [14] we introduce a "generating function" representation of Hermitian matrices and derive all our results in the form of certain polynomial recursions. In Sec 4 we introduce the notions of *block generating functions* and *block Quasi-Toeplitz* (BQT) matrices. It is shown that inverse of a QT matrix can be obtained by computing a block Schur complement of a certain associated BQT matrix. Next we derive polynomial recursions for Schur complementation of BQT matrices with regular and arbitrary rank profiles. It must be pointed out that this procedure works under all circumstances including those where a Gohberg-Semencul formula does not hold good. We end this section with an example. Sec 5 contains some concluding remarks.

2 Schur Complementation of Toeplitz and Quasi-Toeplitz Matrices

Since computation of Schur complements is nested, Schur Complements of the leading principal submatrices does not depend on the actual size of the matrix. Hence we can consider without loss of generality that the matrix under consideration is in fact semi-infinite.

Following [14], it is useful to associate M with a "generating" function

$$M(z,w) = [1\ z\ z^2\ \ldots\ldots]M[1\ w\ w^2\ \ldots\ldots]^* \ . \tag{2.1}$$

A Toeplitz matrix of order n $(=1,2,\ldots)$ has the form $T_{n-1} = [t_{i-j}]_{i,j=0}^{n-1}$, where the $\{t_k\}$ are arbitrary $(k = -(n-1),\ \ldots,\ -1,\ 0,\ 1,\ \ldots\ ,(n-1))$. If T_{n-1} is Hermitian then the entries $\{t_k\}_{-(n-1)}^{(n-1)}$ must satisfy $t_{-k} = t_k^*$ for $0 \le k \le n-1$. The semi-infinite extension $T_{n-1,\infty}$ of a finite Hermitian Toeplitz matrix T_{n-1} is the banded semi-infinite Toeplitz matrix with the first row $[t_0^*\ t_1^*\ \ldots\ \ldots\ t_{(n-1)}^*\ 0\ 0\ \ldots\ \ldots]$.

The generating function of this semi-infinite extension can be represented as

$$T_{n-1,\infty}(z,w) = \frac{t(z)+t^*(w)}{1-zw^*} \tag{2.2}$$

where

$$t(z) = \frac{1}{2}t_0 + t_1 z + \ldots + t_{n-1}z^{n-1} \ .$$

Next we shall define a matrix Q to be Quasi-Toeplitz if its generating function $Q(z,w)$ can be represented in the form

$$Q(z,w) = \frac{\alpha(z)\alpha^*(w) - \beta(z)\beta^*(w)}{1-zw} \tag{2.3}$$

where $\alpha(z)$ and $\beta(z)$ are polynomials (in fact they can even be power serieses) in z, say

$$\alpha(z) = \sum_{k=0}^{m} \alpha_k z^k \ \text{ and } \ \beta(z) = \sum_{k=0}^{q} \beta_k z^k \tag{2.4}$$

3 Inversion of Toeplitz and Quasi-Toeplitz Matrices: Block Generating Function Approach

If possible let the Hermitian Toeplitz matrix T_{n-1} of (3.2) be nonsingular. We shall define, the $2n \times 2n$ Toeplitz block matrix

$$T_b = \left[\begin{array}{c|c} T_{n-1} & I \\ I & O \end{array}\right]_{2n\times 2n} \ . \tag{3.1a}$$

T_{n-1} is the first $n \times n$ principal submatrix of T_b and the corresponding block Schur complement is $-T_{n-1}^{-1}$. We define the following semi-infinite matrix

$$T_{b,\infty} = \left[\begin{array}{c|c} T_{n-1,\infty} & I_\infty \\ I_\infty & O_\infty \end{array} \right] . \tag{3.1b}$$

where $T_{n-1,\infty}$ is the semi-infinite extension of T_{n-1} , I_∞ is a semi-infinite identity matrix and O_∞ is a semi-infinite zero matrix. It is useful to associate a semi-infinite block structured matrix $M = \left[\begin{array}{c|c} M_{11} & M_{12} \\ M_{21} & M_{22} \end{array} \right]$. (with structured semi-infinite blocks M_{ij} , $1 \leq i,j \leq 2$) with a block "generating" function

$$M(z,w) = \left[\begin{array}{cccc|cccc} 1 & z & z^2 & \dots & O & \dots & \dots & O \\ O & \dots & \dots & O & 1 & z & z^2 & \dots \end{array} \right] \left[\begin{array}{c|c} M_{11} & M_{12} \\ M_{21} & M_{22} \end{array} \right] \cdot \left[\begin{array}{cccc|cccc} 1 & w & w^2 & \dots & O & \dots & \dots & O \\ O & \dots & \dots & O & 1 & w & w^2 & \dots \end{array} \right]^* \tag{3.2}$$

Then the block generating function (which is obtained by replacing each of the semi-infinite Toeplitz blocks by their generating functions)

$$T_{b,\infty}(z,w) = \left[\begin{array}{c|c} T_{n-1,\infty}(z,w) & \frac{1}{1-zw^*} \\ \frac{1}{1-zw^*} & O \end{array} \right]_{2\times 2} . \tag{3.3}$$

where $T_{n-1,\infty}(z,w)$ is as in (3.2).

For simplicity assume $t_0 = 1$ for now. Then notice that

$$T_{b,\infty}(z,w) = \frac{G_b(z) J G_b^*(w)}{1 - zw^*} \tag{3.4}$$

where

$$J = \left[\begin{array}{c|c} 1 & O \\ O & -1 \end{array} \right] \text{ and } G_b(z) = \left[\begin{array}{c|c} t(z) + \frac{1}{2} & t(z) - \frac{1}{2} \\ 1 & 1 \end{array} \right] .$$

Next we shall define a matrix Q_b to be Block-Quasi-Toeplitz (BQT) if its block generating function $Q_b(z,w)$ can be represented in the form

$$Q_b(z,w) = \frac{1}{1 - zw^*} \left[\begin{array}{c|c} \alpha(z) & \beta(z) \\ \gamma(z) & \delta(z) \end{array} \right] \left[\begin{array}{c|c} 1 & O \\ O & -1 \end{array} \right] \left[\begin{array}{c|c} \alpha(w) & \beta(w) \\ \gamma(w) & \delta(w) \end{array} \right]^* \tag{3.5}$$

where $\alpha(z)$, $\beta(z)$, $\gamma(z)$ and $\delta(z)$ are all polynomials (functions) in z.

Theorem 3.1: The Schur complement of the 1×1 leading principal submatrix of a "BQT" matrix is also BQT.

Proof: The generating function $Q_b^c(z,w)$ of the Schur complement is given by

$$\left[\begin{array}{c|c} z & O \\ O & 1 \end{array} \right] Q_b^c(z,w) \left[\begin{array}{c|c} w^* & O \\ O & 1 \end{array} \right] = Q_b(z,w) - Q_b(z,0)[1\ 0]^*([1\ 0]Q_b(0,0)[1\ 0]^*)^{-1}[1\ 0]Q_b(0, \tag{3.6}$$

where $Q_b(z,w)$ in (3.6) is the block generating function of BQT matrix Q_b. Assuming that $Q_b(0,0) \neq 0$ we obtain after some algebra

$$Q_b^c(z,w) = \frac{\left[\begin{array}{c|c} \alpha_1(z) & \beta_1(z) \\ z\gamma_1(z) & \delta_1(z) \end{array} \right] \left[\begin{array}{c|c} 1 & O \\ O & -1 \end{array} \right] \left[\begin{array}{c|c} \alpha_1(w) & \beta_1(w) \\ w\gamma_1(w) & \delta_1(w) \end{array} \right]^*}{(|\alpha(0)|^2 - |\beta(0)|^2)(1 - zw^*)} \tag{3.7a}$$

where

$$\alpha_1(z) = \alpha(z)\alpha^*(0) - \beta(z)\beta^*(0) \quad , \gamma_1(z) = \gamma(z)\alpha^*(0) - \delta(z)\beta^*(0) \quad ,$$

$$\beta_1(z) = [\alpha(z)\beta(0) - \beta(z)\alpha(0)]/z, \quad \text{and} \quad \delta_1(z) = [\gamma(z)\beta(0) - \delta(z)\alpha(0)] \; . \qquad (3.7b)$$

It is clear that $\beta_1(z)$ is a polynomial since the numerator in (3.7b) has a root at the origin. This completes the proof of the theorem, since $Q_b^c(z,w)$ is clearly BQT. □

It is clear that n steps of Schur reduction on the block generating function $T_{b,\infty}(z,w)$ would directly lead to the generating function of T_{n-1}^{-1} via the computed polynomials (functions) $\gamma_{n-1}(z)$ and $\delta_{n-1}(z)$ starting with $\gamma_0(z) = 1$ and $\delta_0(z) = 1$ as in (3.4).

It has been shown in [15] (see [15] appendix A, sec. A.2, equation (A.5)) that for an arbitrary nonsingular Quasi-Toeplitz matrix Q_{n-1} it is $Q_{n-1}^{-\natural}$ [4] and not Q_{n-1}^{-1} that posses the same displacement rank. This naturally raises the question: is it possible to compute $Q_{n-1}^{-\natural}$ in a similar fashion. Given $\alpha(z)$ and $\beta(z)$ it is possible to find $\gamma_e(z)$ such that

$$[\alpha(z) - \beta(z)]\gamma_e(z) = 1 + O(z^n) \; . \qquad (3.8a)$$

Partition $\gamma_e(z)$ as $\gamma_e(z) = \hat{\gamma}_e(z) + O(z^n)$. Then the choice

$$\gamma(z) = \delta(z) = \hat{\gamma}_e(z) \qquad (3.8b)$$

would be sufficient and the Schur procedure would compute the generators for $Q^{-\natural}$ in n steps.

3.1 Singularities and Block Schur Complements

The Schur complement $Q_b^c(z,w)$ of the 1×1 leading principal submatrix of a "BQT" matrix can not be computed if $[1 \; 0]Q_b(0,0)[1 \; 0]^* = Q(0,0) = 0$ (see (3.6)).

3.1.1 Immittance Variables and the Structure of Q_b

It is convenient to understand singularities in terms of the immittance variables (see [16]). Motivated by this we rewrite the expression (3.5) using the following so called immittance variables

$$\left.\begin{array}{rcl} \sqrt{2}a(z) & = & \alpha(z) - (\alpha(0)/\beta(0))\beta(z) \\ \sqrt{2}b(z) & = & \alpha(z) + (\alpha(0)/\beta(0))\beta(z) \\ \sqrt{2}c(z) & = & \gamma(z) - (\alpha(0)/\beta(0))\delta(z) \\ \sqrt{2}d(z) & = & \gamma(z) + (\alpha(0)/\beta(0))\delta(z) \end{array}\right\} \qquad (3.9)$$

After some algebra we get

$$\begin{array}{rcl} Q_b(z,w) & = & \dfrac{\left[\begin{array}{c|c} a(z)b^*(w) + b(z)a^*(w) & a(z)d^*(w) + b(z)c^*(w) \\ a^*(w)d(z) + b^*(w)c(z) & c(z)d^*(w) + d(z)c^*(w) \end{array}\right]}{(1-zw^*)} \\ & = & \dfrac{\left[\begin{array}{c|c} a(z) & b(z) \\ c(z) & d(z) \end{array}\right]\left[\begin{array}{c|c} O & 1 \\ 1 & O \end{array}\right]\left[\begin{array}{c|c} a(w) & b(w) \\ c(w) & d(w) \end{array}\right]^*}{(1-zw^*)} \\ & = & \left[\begin{array}{c|c} Q(z,w) & Q_X^*(w,z) \\ Q_X(z,w) & Q_I(z,w) \end{array}\right] \quad \text{say.} \end{array} \qquad (3.10)$$

---

[4] $Q^\natural = \tilde{I}Q^T\tilde{I}$, is sometimes called Q-natural; $\tilde{I}$ being the reversed identity matrix and T being the transpose.

Let $a(z) = z^t a_t(z)$ Then it can be shown that the $2t \times 2t$ block Schur complement Q_I^c of Q_I must also be BQT viz.

$$Q_I^c(z,w) = \frac{\left[\begin{array}{c|c} a_t(z) & b_t(z) \\ c_t(z) & d_t(z) \end{array}\right]\left[\begin{array}{c|c} O & 1 \\ 1 & O \end{array}\right]\left[\begin{array}{c|c} a_t(w) & b_t(w) \\ c_t(w) & d_t(w) \end{array}\right]^*}{(1-zw^*)} \quad \text{say.} \tag{3.11}$$

Where one recursively computes $c_t(z)$ and $d_t(z)$ using the $\{\tau_j\}$ in (3.8b) using the above recursion

$$\left[\begin{array}{c|c} z & O \\ O & z^{-1} \end{array}\right]\left[\begin{array}{c|c} a_{j+1}(z) & b_{j+1}(z) \\ c_{j+1}(z) & d_{j+1}(z) \end{array}\right] = \left[\begin{array}{c|c} a_j(z) & b_j(z) \\ c_j(z) & d_j(z) \end{array}\right]\left[\begin{array}{c|c} 1 & -\tau_j z^{-(t-j)} + \tau_j^* z^{t-j} \\ O & 1 \end{array}\right]. \tag{3.12}$$

3.2 Polynomial Recursions for Schur Complementation

We can rewrite (3.7a) as follows

$$\begin{aligned} Q_b^c(z,w) &= \frac{\left[\begin{array}{c|c} \alpha_1(z) & \beta_1(z) \\ z\gamma_1(z) & \delta_1(z) \end{array}\right]\left[\begin{array}{c|c} 1 & O \\ O & -1 \end{array}\right]\left[\begin{array}{c|c} \alpha_1(w) & \beta_1(w) \\ w\gamma_1(w) & \delta_1(w) \end{array}\right]^*}{(|\alpha(0)|^2-|\beta(0)|^2)(1-zw^*)} \\ &= \frac{\left[\begin{array}{c|c} z & O \\ O & 1 \end{array}\right]\left[\begin{array}{c|c} \alpha(z) & \beta(z) \\ \gamma(z) & \delta(z) \end{array}\right]\Theta(z)J\Theta^*(w)\left[\begin{array}{c|c} \alpha(w) & \beta(w) \\ \gamma(w) & \delta(w) \end{array}\right]^*\left[\begin{array}{c|c} w^* & O \\ O & 1 \end{array}\right]}{1-zw^*} \end{aligned} \tag{3.13}$$

where $J = 1 \bigoplus -1$ and the 2×2 matrix $\Theta(z)$ assumes three different forms which are discussed below.

$\Theta(z)$ for a strongly regular step

It is clear that a strongly regular step of Schur complementation is translated into a univariate map $G(z) \rightarrow G(z)\Theta(z)$ of form

$$\left[\begin{array}{c|c} z & O \\ O & 1 \end{array}\right] G_{j+1}(z) = G_j(z)\Theta_j(z) \tag{3.14}$$

Let us define the partitions,

$$G_j(z) = \left[\begin{array}{c|c} \alpha_j(z) & \beta_j(z) \\ \gamma_j(z) & \delta_j(z) \end{array}\right]. \tag{3.15}$$

It should be noted that the transformation $\Theta_j(z)$ is completely determined by $\alpha_j(z)$ and $\beta_j(z)$.
Case 1a: $\alpha_j(0) = 0, \quad |\alpha_j(0)| \neq |\beta_j(0)|$.

This is a strongly regular step (the Schur complement of the (1,1) entry can be computed), for which the recursion is

$$\left.\begin{aligned} z\alpha_{j+1}(z) &= \alpha_j(z) \\ \beta_{j+1}(z) &= \beta_j(z) \end{aligned}\right\} \tag{3.16a}$$

This implies that

$$\Theta_j(z) = \left[\begin{array}{c|c} 1 & O \\ O & z \end{array}\right] \tag{3.16b}$$

Case 1b: $\alpha_j(0) \neq 0,\ |\alpha_j(0)| \neq |\beta_j(0)|$.

This is also a strongly regular step, for which the recursion is

$$\left.\begin{array}{rcl} \alpha_{j+1}(z) & = & \alpha_j(z) - k_j\beta_j(z) \\ z\beta_{j+1}(z) & = & -k_j^*\alpha_j(z) + \beta_j(z) \end{array}\right\} \tag{3.17a}$$

where $k_j = \alpha_j(0)/\beta_j(0)$.

This implies that

$$\Theta_j(z) = \frac{1}{(1-k_jk_j^*)^{\frac{1}{2}}}\left[\begin{array}{c|c} 1 & -k_j \\ -k_j^* & 1 \end{array}\right]\left[\begin{array}{c|c} z & O \\ O & 1 \end{array}\right] \tag{3.17b}$$

if $|k_j| < 1$, otherwise if $|k_j| > 1$ then

$$\Theta_j = \frac{1}{(k_jk_j^* - 1)^{\frac{1}{2}}}\left[\begin{array}{c|c} 1 & -k_j \\ -k_j^* & 1 \end{array}\right]\left[\begin{array}{c|c} z & O \\ O & 1 \end{array}\right] \tag{3.17c}$$

$\Theta(z)$ for a non strongly regular step

Case 2: $\alpha_j(z) = e^{j\phi}\beta_j(z)$, for some $\phi\epsilon[0, 2\pi]$.

Since $Q_j(z,w) = 0$ all the following principal submatrices of Q are singular. Thus at this step there is no need to proceed any further.

Case 3: $\alpha_j(0) = \beta_j(0) = 0$.

Since $Q_j(z,0) = 0 = Q_j(0,w)$, Q_j and all the following principal submatrices are singular and so there is no need to proceed any further.

Case 4: $|\alpha_j(0)| = \beta_j(0)| \neq 0,\ \ \alpha_j(z) \neq e^{j\phi}\beta_j(z)$ for all $\phi\epsilon[0, 2\pi]$.

In this case we need to find a $2t_j \times 2t_j$ block Schur complement provided

$$a_j(z) = \alpha_j(z) - \frac{\alpha_j(0)}{\beta_j(0)}\beta_j(z)$$

is not identically zero and has t_j repeated zero roots. At this point one needs make repeated (t_j times) use of the linear recursion

$$\left.\begin{array}{rcl} za_{j+1}(z) & = & a_j(z) \\ zb_{j+1}(z) & = & b_j(z) - \tau_j z^{-t_j}a_j(z) + \tau_j^* z^{t_j}a_j(z) \\ z^{-1}c_{j+1}(z) & = & c_j(z) \\ z^{-1}d_{j+1}(z) & = & d_j(z) - \tau_j z^{-t_j}c_j(z) + \tau_j^* z^{t_j}c_j(z) \end{array}\right\} \tag{3.18a}$$

where

$$\tau_j = \lim_{z\to 0} z^t b_j(z)/a_j(z)\ .$$

to obtain the generators for the $2t_j \times 2t_j$ block Schur complement. This implies that the univariate map $G(z) \;\rightarrow\; G(z)\Theta(z)$ assumes the form

$$\left[\begin{array}{c|c} z & O \\ O & z^{-1} \end{array}\right] G_{j+1}(z) = G_j(z)\Theta_j(z) \tag{3.19a}$$

where

$$\Theta_j(z) = \left[\begin{array}{c|c} 1 & -k_j \\ k_j^* & 1 \end{array}\right] \left[\begin{array}{c|c} (1 + \frac{\tau_j^*}{2} z^{t_j} - \frac{\tau_j}{2} z^{-t_j}) & \frac{1}{2}(\tau_j^* z^{t_j} - \tau_j z^{-t_j}) \\ \frac{1}{2}(\tau_j^* z^{t_j} - \tau_j z^{-t_j}) & (1 - \frac{\tau_j^*}{2} z^{t_j} + \frac{\tau_j}{2} z^{-t_j}) \end{array}\right] \tag{3.19b}$$

It is clear that the $\Theta(z)$ in (3.19b) may introduce negative powers of z in the intermediate $\gamma_j(z)$ and $\delta_j(z)$. However the final result would not show any such irregularity. This irregularity can be nicely explained using a transmission line interpretation.

An Example

Next we shall consider an inverting a 3×3 nonsingular symmetric Toeplitz matrix whose 2×2 principal minor is zero. Consider the 3×3 symmetric Toeplitz matrix T with first row [1 1 0]. The starting generator

$$G_0(z) = \left[\begin{array}{c|c} \alpha_0(z) & \beta_0(z) \\ \gamma_0(z) & \delta_0(z) \end{array}\right] = \left[\begin{array}{c|c} 1+z & z \\ 1 & 1 \end{array}\right] . \tag{3.20a}$$

Using (3.17a)-(4.17b) we get

$$k_0 = 0 \text{ and } \Theta_0(z) = \left[\begin{array}{c|c} z & O \\ O & 1 \end{array}\right]$$

which implies

$$G_1(z) = \left[\begin{array}{c|c} \alpha_1(z) & \beta_1(z) \\ \gamma_1(z) & \delta_1(z) \end{array}\right] = \left[\begin{array}{c|c} z + z^2 & z \\ z & 1 \end{array}\right] . \tag{3.20b}$$

Then using (3.19a)-(4.19b) we get

$$k_1 = 1 \ , \ t_1 = 1 \text{ and } \Theta_1(z) = \left[\begin{array}{c|c} (1 + z - z^{-1}) & (z - z^{-1}) \\ -(z - z^{-1}) & (1 - z + z^{-1}) \end{array}\right]$$

which implies

$$\left[\begin{array}{c|c} \gamma_3(z) & \delta_3(z) \end{array}\right] = \left[\begin{array}{c|c} z^{-1} - 1 + z^2 & z^{-1} - z + z^2 \end{array}\right] .$$

The causal generator for negative of the inverse of T must be $[z\gamma_3(z) \;\; z\delta_3(z)] = [1 - z + z^3 \;\; 1 - z^2 + z^3]$. A straight forward calculation verifies this claim. In this particular case it is not possible to find a Gohberg-Semencul formula but our procedure provides us with one similar kind of formula.

4 Concluding Remarks

A new approach that provides a unified framework for fast and completely recursive procedures for computing inverse of Hermitian Toeplitz and Quasi-Toeplitz matrices with arbitrary rank profile has been developed. This recursive procedure is an extension of Chun's [3] results on positive definite Toeplitz matrices. Our approach is based on the fact that the Schur complement (or the block Schur complement) of the top left entry (or block) of a Block Toeplitz or a Block Quasi-Toeplitz (BQT) matrix is Block Quasi-Toeplitz; except for Chun [3] other authors did not make use of this fact. We have also shown how to adopt this procedure to obtain solution to linear equations with nonsingular Quasi-Toeplitz coefficient matrices.

This procedure does not require computation of inner products and is completely parallelizable. Determination of the size of a block factor is done by counting the number of repeated zeros at the origin of a polynomial.

The inversion procedures work as long as the underlying matrices are nonsingular. Unlike all previous approaches (see e.g. Heinig and Rost [1] and Ben-Artzi and Shalom [17]) no special requirement on the rank profile is needed.

Since theoritical detection of a singularity and the number of consecutive zero minors require infinite precision, it is clear that a floating point implementation of the algorithm would contain serious problems. This raises the issue of devising a numerically sound general algorithm which would be able to deal with "nearly singular cases". Such a question goes beyond the scope of the paper and would deserve a thorough investigation.

Acknowledgements

The authors would also like to thank Professor Hanoch Lev-Ari of the Northeastern University, for his suggestions for improvement and constructive criticism. The first author would like to thank Dr. Victor B. Lawrence and Dr. Joseph G. Kneuer of the AT&T Bell Laboratories for encouragement and support.

References

[1] G. Heinig and K. Rost. "Algebraic Methods for Toeplitz-like Matrices and Operators". *Akademie-Verlag, Berlin*, 1984.

[2] T. Kailath. "A Theorem of I. Schur and Its Impact on Modern Signal Processing". *I. Schur Methods in Operator Theory and Signal Processing*, 1986.

[3] Joowhan Chun. *"Fast Array Algorithms for Structured Matrices"*. PhD thesis, Stanford University, Stanford, CA, June 1989.

[4] H. Lev-Ari, Y. Bistritz, and T. Kailath. "Generalized Bezoutians and Families of Efficient Zero-Location Procedures". *to appear in IEEE Transactions on Circuits and Systems.*, 1990.

[5] M.G. Krein and M.A. Naimark. "The Method of Symmetric and Hermitian Forms in the Theory of the Seperation of Roots of Algebraic Equations (Originally in Russian, Kharkov 1936)". *Linear and Multilinear Algebra*, 10:265–308, 1981.

[6] Y. Bistritz and T. Kailath. "Inversion and Factorization of Non-Hermitian Quasi-Toeplitz Matrices". *Linear Algebra and Its Applications*, 98:77–121, Jan. 1988.

[7] I. Gohberg, T. Kailath, and I. Koltracht. "Efficient Solution of Linear Systems of Equations with Recursive Structure". *Linear Algebra and Its Applications*, 80:81–113, 1986.

[8] M.J.C. Gover and S. Barnett. "Inversion of Toeplitz matrices which are not strongly nonsingular". *IMA Journal of Numerical Analysis*, 5:101–110, 1985.

[9] G. Labahn, D.K. Choi, and S. Cabay. "The Inverses of Block Hankel and Block Toeplitz Matrices". *SIAM Journal on Computing*, 19:93–123, Feb. 1990.

[10] J. Rissanen. "Solution of Linear Equations with Hankel and Toeplitz Matrices". *Numer. Math. 22, 367-361*, 22:361–366, 1974.

[11] P. Delsarte, Y. Genin, and Y. Kamp. "A Generalization of the Levinson Algorithm for Hermitian Toeplitz Matrices with Arbitrary Rank Profile". *IEEE Transactions on ASSP*, ASSP.-33:964–971, 1985.

[12] Y. Sugiyama. "An Algorithm for Solving Discrete Time Wiener-Hopf Equations Based on Euclid's Algorithm". *IEEE Transactions on Information Theory*, IT-32:394–409, 1986.

[13] S. Pombra, H. Lev-Ari, and T. Kailath. "Levinson and Schur Algorithms for Toeplitz Matrices with Singular Minors". *Proc. ICASSP-88, New York*, pages 1643–1646, 1988.

[14] H. Lev-Ari and T. Kailath. "Triangular Factorization of Structured Hermitian Matrices". *Operator Theory: Advances and Applications*, 18:301–324, 1986.

[15] D. Pal. *Fast Algorithms for Structured Matrices with Arbitrary Rank Profile.* PhD thesis, Stanford University, Stanford, CA, May. 1990.

[16] D. Pal and T. Kailath. "Factorization of Hermitian Toeplitz and Related Matrices with Arbitrary Rank Profile". *Submitted to SIAM Journal on Matrix Analysis and Applications*, 1989.

[17] A. Ben-Artzi and T. Shalom. "On Inversion of Toeplitz and Close to Toeplitz matrices". *Linear Algebra and its Applications*, 75:173–192, 1986.

ITERATIVE ALGORITHMS FOR COMPUTING STRUCTURED RANK-REDUCING MATRIX PERTURBATIONS

Mark Wicks and Raymond DeCarlo
School of Electrical Engineerng
Purdue University
West Lafayette, Indiana 47907

1.0 Introduction

The problem of determining structured rank-reducing matrix perturbations for general matrices arises frequently in control theory (as well as many other engineering disciplines). A rank-reducing matrix perturbation is any matrix from an allowable perturbation set which, when added to some given matrix, produces a rank-deficient matrix. The eigenvalue problem is probably the most familiar example of this problem: given a square matrix, A, find a matrix having the form λI for which $\det(A - \lambda I) = 0$. The generalized eigenvalue problem is also an example of the problem of finding a structured rank-reducing matrix perturbation: given square matrices, A and B, find a matrix having the form λB for which $\det(A - \lambda B) = 0$. Rectangular matrix problems arise as well. Consider the problem of finding transmission zeros: given the matrices A, B, and C, find λ so that

$$\det \begin{pmatrix} A & B \\ C & 0 \end{pmatrix} - \lambda \begin{pmatrix} I & 0 \\ 0 & 0 \end{pmatrix} = 0.$$

Multiparameter problems can arise from some boundary value problems. For example, consider the eigentuple problem: given A and B_i, find λ_i and $x \neq 0$ for which

$$(A - \sum_{i=1}^{n} \lambda_i B_i)x = 0.$$

All of these problems are special cases of the rank-reducing perturbation problem (RRP) as given in Definition 1.1.

Definition 1.1 *The rank-reducing perturbation problem (RRP).* Given $M \in \mathcal{C}^{n \times m}$ with $n \leq m$ and a vector space $\mathcal{R} \subset \mathcal{C}^{n \times m}$ determine $R \in \mathcal{R}$ (if possible) so that $\operatorname{rank}(M - R) < n$.

The coefficient field for the vector space $\mathcal{R}$ may be $I\!R$ or $\mathcal{C}$. Any R satisfying definition 1.1 will be called a rank-reducing perturbation (RRP).

Methods for determining numerical solutions for various special cases of this problem have been well-developed, especially for the case when the dimension of $\mathcal{R}$ is one and M is square. For example, many algorithms exist for solution of the eigenvalue problem. Numerical algorithms for the generalized eigenvalue problem have also been well-developed. Among the more popular techniques are the QZ algorithm and Rayleigh iteration (See for example, [7]). The QZ algorithm simultaneously reduces both A and B to upper triangular form. In this form all solutions are obtained simultaneously. When the matrices involved are symmetric, Rayleigh iteration, will rapidly converge to a single solution.

Theory for the non-square case when the dimension of $\mathcal{R}$ is one is abundant. In this case, the RRP is equivalent to determining the singularity of a rectangular matrix pencil. A pencil can be reduced to a canonical form, the so-called Kronecker form, in which the solution is apparent. It is not apparent that such a canonical form exists for the multiparameter problems.

Algorithms for the case when $\mathcal{R}$ is multidimensional are under development. Under certain conditions the parameters can be decoupled and the problem can be reduced to two generalized eigenvalue problems [8]. In this case conventional eigenvalue techniques can be used. In [1] and [2], iterative gradient methods are explored for the general RRP. These techniques iteratively update x in the direction of the gradient of the function

$$F(x) = \frac{1}{2}\|Ax - \sum_{i=1}^{r} \lambda_i(x) B_i x\|^2.$$

The $\lambda_i(x)$ are defined in such a way as to minimize the objective function for each fixed x. The papers remark that this technique is rather slow; in effect, this is a gradient minimization of $F(x)$.

For the general rectangular RRP, there may be no solution. Moreover, in many applications, the existance of an RRP is undesirable. For example, consider the usual linear time-invariant state model,

$$\dot{x}(t) = Ax(t) + Bu(t),$$

where $A \in \mathbb{C}^{n\times n}$ and $B \in \mathbb{C}^{n\times r}$. The existance of a $\lambda \in \mathbb{C}$ for which $\text{rank}[\, A - \lambda I, \quad B\,] < n$ indicates uncontrollability. This is often undesirable. If $\text{rank}[\, A - \lambda I, \quad B\,] = n$ for all $\lambda \in \mathbb{C}$, one may wish to find the value of λ (there exists one, see [9]) for which the matrix most nearly fails to have full rank, i.e., that value of λ which minimizes $\sigma_n([\, A - \lambda I, \quad B\,])$ as well as the corresponding perturbation to the matrices A and B which would result in uncontrollability ($\sigma_n(M)$ denotes the n^{th} singular value of M satisfying $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$). The problem of determining such a perturbation is called the controllability robustness problem. For discussions of this problem see [5], [6], and [9].

Similarly, the existance of a $\lambda \in \mathbb{C}$ for which $\text{Re}[\lambda] \geq 0$ and $\text{rank}[A - \lambda I] < n$ indicates instability, or more precisely, failure of the system to be asymptotically stable. In this case, one may wish to know that value of ω which minimizes $\sigma_n([A - j\omega I])$ as well as the corresponding perturbation to the matrix A which results in instability. The problem of determining such a perturbation is called the stability robustness problem. Stability robustness is discussed, for example, in [4] and [10]

These problems are special cases of a more general problem which will be designated the almost rank-reducing perturbation problem (ARRP). Definition 1.2 formalizes this problem.

Definition 1.2 *The almost rank-reducing perturbation problem (ARRP).* Given $M \in \mathbb{C}^{n\times m}$ with $n \leq m$ and a vector space $\mathcal{R} \subset \mathbb{C}^{n\times m}$ determine $R \in \mathcal{R}$ (if one exists) which minimizes $\sigma_n(M - R)$.

The coefficient field for the vector space $\mathcal{R}$ may be $\mathbb{R}$ or $\mathbb{C}$. This minimizing R is called an almost rank-reducing perturbation (ARRP). It will be useful to define a measure of the size of the almost rank-reducing perturbation. This will be called the rank-robustness of M along $\mathcal{R}$ as in Definition 1.3.

Definition 1.3 Given $M \in \mathbb{C}^{n\times m}$ with $n \leq m$ and a vector space $\mathcal{R} \subset \mathbb{C}^{n\times m}$ define *the rank-robustness of M along $\mathcal{R}$* as

$$r(M;\, \mathcal{R}) \stackrel{\Delta}{=} \inf_{R\in\mathcal{R},\text{rank}[M-R]<n} \|R\|$$

where the norm used is the Frobenius matrix norm.

Note that any RRP is also an ARRP so a method for solving the ARRP problem will also solve the RRP problem. The reverse does not hold. Thus, a single method which solves the ARRP (and hence the RRP) is desirable.

Numerical methods have been developed for several special cases of the ARRP. The one dimensional case when the coefficient field for R is $\mathbb{C}$ is considered in [3]. Boley's method is to choose $R \in \mathcal{R}$ which minimizes $\|u^H M - u^H R\|$ at each iteration (with u fixed) and then update u to be a left singular vector associated with $\sigma_n(M - R)$. This method is globally convergent (but not necessarily to a global minimizer) and easily generalized to several dimensions. The main disadvantage with this technique is the large amount of work required per iteration compounded by a slow rate of convergence. Similar techniques are considered in [9] for the controllability robustness problem.

The stability robustness problem is a special case of the ARRP when the coefficient field for R is $\mathbb{R}$ and the dimension of R is one. A bisection method for finding solutions for the stability robustness problem appears in [4]. It has the advantages of being convergent to a global minimum and converging in a finite number of steps. The number of steps is known a priori. It is not obvious how to generalize the technique if the dimension of $\mathbb{R}$ is greater than one or if the coefficient field of $\mathbb{R}$ is $\mathbb{C}$. Briefly the technique works as follows: first a region is determined that bounds the minimum value of $\sigma_n(M - R)$ for $R \in \mathbb{R}$; secondly, the midpoint of the region is tested to see if there are any $R \in \mathbb{R}$ for which the value of $\sigma_n(M - R)$ is equal to the midpoint (this test involves the solution of a simple eigenvalue problem); lastly, the result of this test allows one to bisect the region and identify the half containing the solution. This process is repeated until the interval containing the solution achieves some predetermined size.

This paper discusses a class of algorithms for solving the RRP and the ARRP. The approach taken is to view a RRP as a special case of an ARRP. Algorithms based on this approach attempt to drive the n^{th} singular value of $M - R$ toward zero. From this perspective, singular points are not viewed as isolated points, but as zeros of the surface defined by $\sigma_n(M - R)$ as R varies over $\mathcal{R}$. First, the paper discusses a Newton method of reducing the smallest singular value of $M - R$ to zero to find an RRP. Such a method will fail near a local minimum of $\sigma_n(M - R)$ as the derivative of $\sigma_n(M - R)$ vanishes. To avoid this difficulty, an algorithm is proposed which will always converge to either a RRP or an ARRP (a local minimum) while retaining the convergence rate of the Newton method to an RRP. Convergence to an ARRP is nearly as rapid. Some numerical results obtained from executing the algorithms on several examples will be presented.

Before discussing these algorithms some elementary properties of the problems introduced above will be useful. Section 2.0 will discuss some theoretical issues concerning the existance of solutions to the problems and the asymptotic behavior of $\sigma_n(M - R)$ as $\|R\| \to \infty$. Section 3.0 will discuss the algorithms. Section 4.0 will present some numerical results.

2.0 Properties of the Problems

This section discusses some elementary properties concerning existance of solutions to the problems introduced above.

Property 2.1 For a given matrix, M, there may be no finite RRP.

Numerous examples appear for the generalized eigenvalue problem. Moreover, there may be no infinite RRP. This will be a consequence of the definition of an infinite RRP appearing later in this section.

The asymptotic behaviour of $\sigma_n(M - R)$ as $\|R\| \to \infty$ helps to understand the global behaviour of the singular value surfaces and can provide insight into determining suitable starting values for numerical algorithms. The next property and its corollaries characterize the asymptotic behavior of singular values as functions of matrices in $\mathcal{R}$.

Property 2.2 Given M and $R \in \mathcal{R}$ as before, suppose the first n singular values of R are distinct. As $\alpha \longrightarrow \infty$, $\alpha \in I\!R$, the asymptote of $\sigma_n(M - \alpha R)$ is given by

$$| - \mathrm{Re}[u^H M v] + \alpha\sigma_n(R)|$$

where u and v are some pair of left and right unit length singular vectors of R associated with $\sigma_n(R)$.

Corollary 2.3 If the singular values of R are non-zero for all $R \in \mathcal{R}$ (this requires $\mathcal{R}$ be one-dimensional) then a finite solution exists to the ARRP.

Corollary 2.4 $\lim_{\alpha\rightarrow\infty} \sigma_n(M - \alpha R)$ is finite if and only if $\mathrm{rank}[R] < n$.

Before presenting the next group of properties, it is necessary to distinguish between finite solutions to the RRP and infinite solutions to the RRP. The RRP has a finite solution if and only if some $R \in \mathcal{R}$ satisfies $\sigma_n(M - R) = 0$. The RRP is said to have an infinite solution if for some $R \in \mathcal{R}$

$$\lim_{\alpha\longrightarrow\infty} \sigma_n(M - \alpha R) = 0.$$

Similarly, the ARRP is said to have an infinite solution if for some $R_0 \in \mathcal{R}$

$$\lim_{\alpha\longrightarrow\infty} \sigma_n(M - \alpha R_0) = \inf_{R\in\mathcal{R}} \sigma_n(M - R).$$

Using these definitions, one can immediately deduce the next corollary from Property 2.2.

Corollary 2.5 The ARRP always has a solution, which may be a finite or an infinite solution.

The previous corollary reinforces the desirability of computing ARRP solutions rather than RRP solutions. The former always exists while the latter may not exist in any sense, finite or infinite.

Finally a complete characterization of infinite solutions to the RRP is possible:

Proposition 2.6 Given M and $\mathcal{R}$ as before, an infinite solution to the RRP exists if and only if for some $R \in \mathcal{R}$

$$\mathrm{rank}\begin{pmatrix} M & R \\ R & 0 \end{pmatrix} < n + \mathrm{rank}[R] \tag{2.7}$$

where n is the row dimension of M and R.

Proof. Suppose for some $R \in \mathcal{R}$ (2.7) holds. First, it is necessary to establish the existance of $x, y \in \mathcal{C}^n$, $x \neq 0$ for which $x^H M + y^H R = 0$ and $x^H R = 0$. To do this consider a basis for the left null-space of R, say $\{y_i\}$, $i = 1, \ldots, n - \mathrm{rank}[R]$. The set of vectors $\{\mathrm{col}[\,0, \quad y_i\,]\}$, $i = 1, \ldots, n - \mathrm{rank}[R]$ forms a linearly independent set contained in the left null-space of

$$\begin{pmatrix} M & R \\ R & 0 \end{pmatrix}.$$

But (2.7) implies the dimension of this left null-space is greater than $n - \mathrm{rank}[R]$. This implies the existance of another vector, $\mathrm{col}[\,x, \quad y\,]$, in this left null-space that is linearly independent of the set $\{\mathrm{col}[\,0, \quad y_i\,]\}$, $i = 1, \ldots, n - \mathrm{rank}[R]$. It must be that $x \neq 0$; otherwise one would obtain $y^H R = 0$ with y being linearly independent of the set of $\{y_i\}$, which cannot be.

Using the vectors x and y obtained from $\mathrm{col}[\,x, \quad y\,]$ above, consider the product

$$(x + (1/\alpha)y)^H (M + \alpha R) = (1/\alpha) y^H M.$$

Taking the limit of this product as $\alpha \to \infty$ yields zero, i.e.,

$$\lim_{\alpha \to \infty} (x + (1/\alpha)y)^H (M + \alpha R) = 0.$$

Since $x \neq 0$ and since $n \leq m$, this implies that

$$\lim_{\alpha \to \infty} \sigma_n(M + \alpha R) = 0,$$

satisfying the definition of an infinite RRP solution.

Conversely, suppose for some $R \in \mathcal{R}$

$$\lim_{\alpha \to \infty} \sigma_n(M + \alpha R) = 0.$$

This implies the existance of sequences, α_k and z_k for which $\alpha_k \to \infty$, $||z_k|| = 1$ and $||z_k^H(M + \alpha_k R)|| \to 0$. The vector z_k can be written (in a unique way) as $z_k = x_k + (1/\alpha_k)y_k$ for some $x_k \in \ker[R^H]$ and $y_k \in \mathrm{range}[R]$. From this construction it follows that $||x_k|| \leq 1$ and $||(1/\alpha_k)y_k|| \leq 1$. Using these vectors, one obtains

$$||(x_k + (1/\alpha_k)y_k)^H(M + \alpha_k R)|| = ||x_k^H M + y_k^H R + (1/\alpha_k)y_k^H R|| \to 0. \tag{2.8}$$

It follows from (2.8) that $||y_k||$ is bounded; otherwise $||y_k^H R||$ would be unbounded since $y_k \in \mathrm{range}[R]$ and (2.8) would not converge to zero since all other terms are obviously bounded as described above. Moreover, it follows that $x_k^H M + y_k^H R \to 0$ since boundedness of $||y_k||$ implies $(1/\alpha_k)y_k \to 0$.

Also, it follows that $||x_k|| \to 1$ since $||y_k||$ is bounded and $||z_k||^2 = ||x_k||^2 + (1/\alpha)^2||y_k||^2 = 1$. Thus, arbitrary cluster points of the sequences, x_k and y_k, say x_* and y_*, must satisfy $x_*^H M + y_*^H R = 0$, $x_*^H R = 0$, and $x_* \neq 0$, i.e., $\mathrm{col}[\, x_* \quad y_* \,]$ is contained in the left null-space of

$$\begin{pmatrix} M & R \\ R & 0 \end{pmatrix}. \tag{2.9}$$

Again, let $\{y_i\}$, $i = 1, \ldots, n - \mathrm{rank}[R]$ be a basis for the left null-space of R. Clearly the vectors, $\{\mathrm{col}[\, 0, \quad y_i \,]\}$ are contained in the left null-space of the matrix in (2.9) and are linearly independent of $\mathrm{col}[\, x_*, \quad y_* \,]$ because $x_* \neq 0$, i.e., there exist $n - \mathrm{rank}[R] + 1$ linearly independent vectors in the left null-space of the matrix given in (2.9). This implies (2.7).

Depending on the dimension and structure of the matrices involved, the nonexistance of a solution to the RRP may be generic, i.e., even though a solution to the RRP may exist in principle, it may not exist, numerically speaking. Numerically, the RRP problem can be perturbed into an ARRP problem. This problem is discussed in [9] within the context of the controllability problem mentioned earlier. This provides additional motivation for designing single algorithms which work on either problem. Such an algorithm will be developed in the following section.

3.0 Algorithm Development

A Newton approach to the problem is formulated by differentiating the smallest singular value of $M - R$. Singular values are not differentiable at points where they are non-distinct. In practice, this poses no difficulty because the set of points where the singular values are non-distinct has lower dimension than $\mathcal{R}$ for many problems of interest. Certain problem structures may cause the singular values to be non-distinct throughout $\mathcal{R}$. This paper will not consider such problem structures. Henceforth, this paper assumes that the singular values of $M - R$ are distinct for almost all R in $\mathcal{R}$.

The function

$$f(\alpha) \stackrel{\Delta}{=} \sigma_n(M - \alpha R) \tag{3.1}$$

is differentiable with respect to α (α is real) as long as the singular values of $M - R$ are distinct. The derivative of f is given by $-\operatorname{Re}[u^H Rv]$ where u and v form a left and right unit length singular vector pair associated with $\sigma_n(M - \alpha R)$. More specifically, the pair u, v satisfies

$$\begin{aligned} (M - \alpha R)v &= \sigma_n(M - \alpha R)u \\ (M - \alpha R)^H u &= \sigma_n(M - \alpha R)v. \end{aligned} \tag{3.2}$$

When M and all $R \in \mathcal{R}$ are real this differentiation poses no difficulty and there results a simple expression for Newton iteration for finding a RRP. For example, when $\mathcal{R}$ is a one-dimensional space having R as its basis, one obtains the formulation,

$$\begin{aligned} \alpha_{k+1} &= \alpha_k + [u_k^H(M - \alpha_k R)v_k]/(u_k^H R v_k) \\ &= (u_k^H M v_k)/(u_k^H R v_k) \end{aligned} \tag{3.3}$$

where u_k and v_k are left and right unit length singular vectors associated with $\sigma_n(M - \alpha_k R)$ as in (3.2). † The sequence $\alpha_k \to \alpha_*$ as $k \to \infty$ and the quantity $\alpha_* R$ becomes the RRP.

When $\mathcal{R}$ is multi-dimensional, one must locate $\Delta R_k \in \mathcal{R}$ such that

$$u_k^H(\Delta R_k)v_k = u_k^H(M - R_k)v_k \tag{3.4}$$

and set $R_{k+1} = R_k + \Delta R_k$.

This follows because a Newton iteration sets

$$0 = \sigma_n + \Delta\sigma_n = u^H(M - R)v - u^H(\Delta R)v$$

which is equivalent to (3.4).

On the other hand, when M or R have non-zero imaginary parts, the successive approximations of the Newton iteration are determined by locating $\Delta R_k \in \mathcal{R}$ such that

$$\operatorname{Re}[u_k^H(\Delta R_k)v_k] = u_k^H(M - R_k)v_k \tag{3.5}$$

with $R_{k+1} = R_k + \Delta R_k$. This formulation is more difficult to implement than (3.4) because of the presence of the real part. The real part is present because the derivative of the function in 3.1 is $\operatorname{Re}[u^H Rv]$ duly generalized to multiple dimensions in (3.5)

The first main result of this section shows that the quantity $-u^H Rv$ may be substituted for the derivative in a Newton iteration when M or R are complex. Strictly speaking, the quantity $-u^H Rv$ is not the derivative of (3.1), but it is the derivative of a complex valued function having the same modulus as $\sigma_n(M - \alpha R)$. Obviously locating zeros of this function is equivalent to locating zeros of $\sigma_n(M - \alpha R)$. This result will justify using the iteration given by (3.4) even when R and M are complex.

The following characterization of singular values will prove useful in the subsequent discussion:

Proposition 3.6 Let $M \in \mathcal{C}^{n\times m}$ and $c \in \mathcal{C}$, then $|c| = \sigma_n(M)$ if and only if there exist $y \in \mathcal{C}^m$ and $z \in \mathcal{C}^n$ not both zero for which

$$\begin{aligned} Mz &= cy \\ M^H y &= \bar{c}z. \end{aligned} \tag{3.7a}$$

---

† Equation (3.3) is noticeably similar to Rayleigh quotient iteration [7] and when $M - \alpha_k R$ is symmetric, it is virtually the same. This follows because $M - \alpha_k R$ being symmetric implies $u_k = \pm v_k$.

In matrix form, (3.7a) may be expressed as

$$\begin{pmatrix} -cI & M \\ M^H & -\bar{c}I \end{pmatrix} \begin{pmatrix} y \\ z \end{pmatrix} = 0 \tag{3.7b}$$

Proposition 3.8 Let $M, R \in \mathcal{C}^{n\times m}$. Given α_0 for which the first n singular values of $M - \alpha_0 R$ are non-zero and distinct, there exists a locally differentiable function, $c(\alpha)$, and functions $y(\alpha)$ and $z(\alpha)$ satisfying (3.7) for which $\dot{c}(\alpha) = -(y^H(\alpha)Rz(\alpha))/(y(\alpha)^H y(\alpha))$. Hence, $|c(\alpha)| = \sigma_n(M - \alpha R)$. Note that α is real.

Proof. Let $u(\alpha)$ and $v(\alpha)$ be a left and right unit length singular vector pair associated with $\sigma_n(M - \alpha R)$. Consider the solution of the initial value problem

$$\dot{c}(\alpha) = -\frac{c(\alpha)}{\sigma_n(M - \alpha R)}(u^H(\alpha)Rv(\alpha))$$

with the initial condition $c(\alpha_0) = \sigma_n(M - \alpha R)$. The solution of this problem exists in a neighborhood of α_0. Clearly

$$\begin{aligned} \frac{d}{d\alpha}c(\alpha)\bar{c}(\alpha) &= -2\frac{c(\alpha)\bar{c}(\alpha)}{\sigma_n(M - \alpha R)}\operatorname{Re}[u^H(\alpha)Rv(\alpha)] \\ &= c(\alpha)\bar{c}(\alpha)\frac{2\dot{\sigma}_n(M - \alpha R)}{\sigma_n(M - \alpha R)} \end{aligned}$$

implying that $c(\alpha)\bar{c}(\alpha) = [\sigma_n(M - \alpha R)]^2$. Suitable functions $y(\alpha)$ and $z(\alpha)$ are guaranteed by Proposition 3.6. It is possible to show that

$$\dot{c}(\alpha) = -(y^H(\alpha)Rz(\alpha))/(y(\alpha)^H y(\alpha))$$

for any $y(\alpha)$ and $z(\alpha)$ satisfying (3.7). This completes the proof.

Clearly $c(\alpha)$ is zero if and only if $\sigma_n(M - \alpha R) = 0$. This suggests applying the Newton method to $c(\alpha)$ rather than applying it to $\sigma_n(M - \alpha R)$. Given any initial guess, say α_0, the derivative of c at α_0 becomes $-u^H(\alpha_0)Rv(\alpha_0)$ since $\sigma_n(M - \alpha_0 R) = c(\alpha_0)$. This amounts to using (3.4) in the complex case as opposed to (3.5).

Iteration based on (3.4) will be unstable near a solution to the ARRP. This occurs near a local minimum of $\sigma_n(M - \alpha R)$ where the derivative vanishes, i.e., $u^H Rv$ approaches zero, making the iteration based on (3.4) unstable. A different approach is suggested by the next proposition.

Proposition 3.9 Let $M \in \mathcal{C}^{n\times m}$, $R_0 \in \mathcal{R}$. Let u and v_n denote a left and right unit length singular vector pair associated with the n^{th} singular value of $M - R_0$. Let $v_{n+1}, \ldots, v_m$ be an orthonormal basis for the right null-space of $M - R_0$. If R_0 is an ARRP then it is necessary that

$$\|u^H(M - R_0)\,[\,v_n \quad v_{n+1} \quad \cdots v_m\,]\,\| = \min_{R\in\mathcal{R}} \|u^H(M - R)\,[\,v_n \quad v_{n+1} \quad \cdots v_m\,]\,\| \tag{3.10}$$

As a means of stabilizing the iteration consider determining the ΔR_k to minimize the least squares problem,

$$\min_{[\Delta R]_k\in\mathcal{R}} \|\|\|[u]_k^H(M - [R]_k - [\Delta R]_k)\,[\,[v_n]_k \quad [v_{n+1}]_k \quad \cdots [v_m]_k\,]\,\| \tag{3.11}$$

and setting $R_{k+1} = R_k + \Delta R_k$. The notation $[\ ]_k$ denotes the k^{th} iterate of the quantity in the brackets. This notation has been introduced to avoid confusion between the i^{th} column vector, v_i and the k^{th} iterate of the i^{th} column vector, $[v_i]_k$. The quantities $[u]_k$ and $[v_i]_k$ are as explained in proposition 3.9 and are singular vectors of $M - [R]_k$. Note that the inversion implied in (3.11) does not necessarily become singular as the quantity $[u]_k^H[R]_k[v_n]_k$ approaches zero. Moreover, solution of (3.11) always yields a descent direction for $f(\alpha)$ as given in (3.1), i.e., there will always be some $R_* \in \mathcal{R}$ on the line connecting $[R]_k$ and $[R]_k + [\Delta R]_k$ for which

$$\sigma_n(M - R_*) < \sigma_n(M - [R]_k)$$

A modified iteration can be used to guarantee global convergence by finding such an R_* and setting $[R]_{k+1}$ to $[R]_*$. However, it is not immediately apparent that the convergence rate of (3.4) is retained locally after including the null-space information in the iteration as in (3.11).

The goal of the remainder of this section is to provide additional rationale for the iteration implied in (3.11) by showing that this iteration is approximately equivalent to the iteration function

$$\begin{aligned}\alpha_{k+1} &= \alpha_k - \frac{f(\alpha_k)}{f'(\alpha_k) + (f(\alpha_k)f''(\alpha_k)/f'(\alpha_k))} \\ &= \alpha_k - \frac{f(\alpha_k)f'(\alpha_k)}{(f')^2(\alpha_k) + f(\alpha_k)f''(\alpha_k)}.\end{aligned} \tag{3.12}$$

This iteration function can converge to either a zero of f or a zero of f'. The iteration function given in (3.12) has the property that it converges quadratically in a neighborhood of a single zero of f. Moreover, near a zero of f' it converges quadratically so long as f and f' do not simultaneously vanish. The iteration specified by (3.11) is an approximation to the iteration of (3.12). Loosely speaking, the approximation becomes better as f becomes smaller so performance will degrade at a solution of the ARRP which does not have a nearly rank-deficient minimum. Performance improves as robustness (as per definition 1.3) decreases.

The correspondance between (3.11) and (3.12) becomes clear after examining the closed form solution of (3.11) in the case that $\mathcal{R}$ is one-dimensional having R as its basis:

$$\alpha_{k+1} = \alpha_k + \frac{([u]_k^H(M - \alpha_k R)[v_n]_k)([u]_k^H R[v_n]_k)}{([u]_k^H R[v_n]_k)^2 + \sum_{i=1}^{m-n}([u]_k^H R[v_{n+i}]_k)^2}. \tag{3.13}$$

Most of the quantities in (3.13) correspond to quantitities in (3.12), i.e., $f(\alpha_k)$ is $[u]_k^H(M - \alpha_k R)[v_n]_k$ and $f'(\alpha_k)$ is $-[u]_k^H R[v_n]_k$. The correspondance with (3.12) would be complete if $f''(\alpha_k)$ were equal to the quantity

$$\frac{1}{\sigma_n(M - \alpha_k R)} \sum_{i=1}^{m-n}([u]_k^H R[v_{n+i}]_k)^2. \tag{3.13}$$

The quantity in (3.13) is a fair approximation to $f''(\alpha_k)$. To see this, consider the second derivative of the quantity $\sigma_n(M - \alpha R)$. If M and R are real, this second derivative may be written as

$$\begin{aligned}\ddot{\sigma}_n(M - \alpha R) &= -(\dot{u}^H Rv + u^H R\dot{v}) \\ &= \begin{pmatrix} u^H & v^H \end{pmatrix} \begin{pmatrix} (u^H Rv)I & -R \\ -R^H & (v^H R^H u)I \end{pmatrix} \begin{pmatrix} \dot{u} \\ \dot{v} \end{pmatrix} =\end{aligned} \tag{3.15}$$

$$-\begin{pmatrix} u \\ v \end{pmatrix}^H \begin{pmatrix} (u^H Rv)I & -R \\ -R^H & (v^H R^H u)I \end{pmatrix} \begin{pmatrix} -\sigma_n(M - \alpha R)I & (M - \alpha R) \\ (M - \alpha R)^H & -\sigma_n(M - \alpha R)I \end{pmatrix}^+ \begin{pmatrix} (u^H Rv)I & -R \\ -R^H & (v^H Ru)I \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}$$

where the quantities u and v are left and right singular vectors associated with $\sigma_n(M - \alpha R)$ (implied to be functions of α). Assuming the singular values of $M - \alpha R$ are distinct, the singular values of the matrix having the pseudo-inverse in (3.15) can be ennumerated as follows: $2n - 2$

non-zero singular values of the form $|\sigma_n \pm \sigma_i|$, two zero singular values, and a singular value equal to σ_n having multiplicity $m - n$. If σ_n is much smaller than the other non-zero singular values then an approximation to the pseudo-inverse can be obtained by truncating the smallest $2n - 2$ non-zero singular values. This results in the approximate equality

$$\begin{pmatrix} -\sigma_n(M - xR)I & (M - \alpha R) \\ (M - xR)^H & -\sigma_n(M - \alpha R)I \end{pmatrix}^+ \approx -\frac{1}{\sigma_n(M - \alpha R)} \sum_{i=1}^{m-n} \begin{pmatrix} 0 & 0 \\ 0 & v_{n+i}v_{n+i}^H \end{pmatrix} \tag{3.16}$$

Substituting this approximation into (3.15) results in the approximate equality

$$\ddot{\sigma}_n(M - \alpha R) \approx \frac{1}{\sigma_n(M - \alpha R)} \sum_{i=1}^{m-n} (u_k^H R v_{n+i})^2.$$

where the v_i, $i = n+1, n+2, \dots, m$ span the null-space of $M - \alpha R$. This demonstrates the approximate equivalence of (3.11) and (3.12) as the condition number of $M - R$ increases. The convergence rate increases as the condition number of $M - R$ increases and the approximation of (3.16) improves.

4.0 Numerical Results

The iteration formula of (3.11) has been applied to several test cases. These numerical experiments demonstrate

i. quadratic convergence to the solution in a neighborhood of an RRP,
ii. rapid linear convergence to the solution of an ARRP if the final robustness measure is small, and
iii. degradation of the linear convergence rate as the robustness measure increases.

Representative results from these test cases appear in Table 4.1. These test cases were generated using MATLAB on a SUN 4 processor. For this example a random real matrix $M \in \mathbb{R}^{10\times 20}$ was generated with entries uniformly distributed between 0 and 1. Four random basis vectors for a real space $\mathcal{R}$ were generated. A matrix M_0 having rank 9 was generated by subracting a rank-reducing perturbation obtained from the singular value decomposition of M. Several random matrices δM having prespecified norms were generated and the iteration given in (3.11) was executed on $M_0 + \delta M$. The rate of convergence r, the associated coefficient of convergence C, and the number of iterations N were experimentally determined. The two smallest non-zero singular values at the final solution point are also tabulated.

$\|\delta M\|$	N	r	C	σ_{10}	σ_9
1	33	0.9985	0.3853	0.1356	0.6726
0.1	6	0.9788	0.0023	0.00610	0.6193
0.01	5	0.9745	0.0007	0.00098	0.6212
0.001	4	-	-	0.00010	0.6204

Table 4.1 - Experimental Results for One Example

The missing table entries were omitted due to an insufficient number of iterations to permit an accurate estimate of their values.

5.0 Conclusion

This paper examines a Newton approach to the solution of the structured rank-reducing perturbation problem as well as the structured almost-rank-reducing perturbation problem. The Newton iteration is based on driving the smallest non-zero singular value of the matrix $M - R$ to zero. The paper demonstrates that a direct implementation of the Newton approach will be unstable near an almost rank-reducing perturbation that is not also a rank-reducing perturbation. The paper proposes a method that stabilizes the Newton iteration, retains quadratic convergence to an RRP, and often has rapid convergence to an ARRP. In addition the paper provides some insight into the asymptotic behavior of singular value surfaces and into the existance of solutions to the RRP and ARRP problems. Some numerical results have been provided to illustrate some of the claims made in the paper.

References

[1] E. K. Blum and P. B. Geltner, "Numerical Solution of Eigentuple-Eigenvector Problems in Hilbert Spaces by a Gradient Method," *Numerische Mathematik*, vol. 31, pp. 247-263, 1978.

[2] E. K. Blum and A. R. Curtis, "A Convergent Gradient Method for Matrix Eigenvector-Eigentuple Problems," *Numerische Mathematik*, vol. 31, pp. 231-246, 1978.

[3] D. Boley, "Computing Rank-Deficiency of Rectangular Matrix Pencils," *System and Control Letters*, vol. 9, pp. 207-214, 1987.

[4] R. Byers, "A Bisection Method for Measuring the Distance of a Stable Matrix to the Unstable Matrices," *SIAM Journal on Scientific and Statistical Computing*, vol. 9, pp. 875-881, 1988.

[5] R. Eising, "The Distance Between a System and the Set of Uncontrollable systems," *Proc. MTNS*, Beer-Sheva, Israel, June 1983, New York: Springer-Verlag, 1984, pp. 303-314

[6] R. Eising, "Between Controllable and Uncontrollable," *Systems and Control Letters*, vol. 4, pp. 263-264, 1984.

[7] G. Golub and C. Van Loan, *Matrix Computations*, Baltimore, MD: Johns Hopkins University Press, 1983.

[8] T. Slivnik and G. Tomšič, "A Numerical Method for the Solution of Two-Parameter Eigenvalue Problems," *Journal of Computational and Applied Mathematics*, vol. 15, pp. 109-115, 1986.

[9] M. Wicks and R. DeCarlo, "Computing the Distance to an Uncontrollable System," *IEEE Transactions on Automatic Control*, AC-36, pp. 39-49, 1991.

[10] C. Van Loan, "How Near is a Stable Matrix to an Unstable Matrix?" *Contemp. Math.*, vol. 47,AMS, Providence, RI, pp. 465-478, 1985.

Signal Reconstruction of Finite Length Sequence Using Initial Time Sample by Hopfield Neural Network

Wenwu Zhu [1] Mouyan Zou [2] Mei Lu [2] Zhenming Chai [2]

[1]Department of Electrical
and Computer Engineering
Illinois Institute of Technology
Chicago, IL 60616

[2]Institute of Electronics,
Academia Sinica,
(Chinese Academy of Sciences)
Beijing, 100080, P R. China

Abstract

Signal reconstruction from its Fourier Transform magnitude(FTM) has been widely investigated[1-3]. Recent results have indicated that when one end point is exactly known, a finite duration sequence can almost always be reconstructed uniquely from its FTM[2]. Although many algorithms were proposed, the complexity of these algorithms is exponential. This phase retrieval problem is NP-complete. In this paper, Hopfield Neural Network(HNN) model is used to solve the phase retrieval problem . The energy function and architecture of Hopfield networks are given. The Boltzmann machine is used to find the minimum of the global energy function. Example is presented, where it is shown that the neural network method is more efficient than the exhaustive search.

Phase Retrieval Problem

The problem of signal reconstruction from its FTM is referred to in the literature as the phase retrieval problem. Supposed x(n) is finite length digital signal with N points. We also assume that we know its FTM and one end point either x(0) or x(N) which is non-zero for a complex signal.

It is easy to see that from the FTM, r(n) may be obtained by the inverse FT. Let R(z) denote the Z-transform of the autocorrelation r(n) of x(n). $R(z)=X(z)X^*(1/z^*)$ $=x(0)x(N)\prod(z-z_i)$ $z_i \varepsilon Z_R^{2(N-1)}$, where $Z_R^{2(N-1)}=Z^x \cup Z^{x^*}$, $Z^x=i\{1/z_1^x, 1/z_2^x, \ldots, 1/z_{(N-1)}^x\}$ and $X^*(1/z^*)$ is inverse conjugate of X(z). Therefore, both z_i and $1/z_i^*$ $(i=1,\ldots,N)$ are zeros of R(z) which appear in pairs. There are 2(N-1) such pairs and it is easy to see that choosing any one of the number of each pair results in a possible solution if only FTM is known.

So the problem of reconstruction signal from FTM can be converted into the problem how can the N-1 zeros of Z^x be correctly chosen among $Z_R^{2(N-1)}$, then, by the chosing correctly Z^x, the original x(n) can be reconstructed. As stated above, if there is no more information than the FTM , there will be 2^{N-1} possible selections for Z^x Thus, end point information must be used to make correct choice. By means of the relationship between the roots and the coefficients of a polynomial, we have: $\prod_{i=0}^{N-1}(-z_i)=x(N-1)/x(0)$, where $r(N-1)=x(0)^* \cdot x(N-1)$

Formation of Hopfield Energy Function and Implementation

Given V_i the output voltage of the ith amplifier, T_{ij} the conductance between the ith and jth amplifiers, and I_i the input bias current to the ith amplifier, the standard Hopfield model is

$$E=-(1/2)\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}T_{ij}V_iV_j-\sum_{i=1}^{N-1}V_iI_i$$

In order to guarantee V_i is digital value which is either 1 or 0, $\sum_{i=0}^{N-1}V_i\,(1-V_i)$ should be one term of energy function which will cause the energy function to favor digital values.

Using the logarithm form, we have $\sum_{i=0}^{N-1}log\,(r_i)=log\,(|x(N-1)/x(0)|)$. Note that phase always automatically satisfies according to the rule chosing the zeros of X(z) with exactly only one member of each pair among 2(N-1). Thus, the computational energy function E of Hopfield Neural Networks is formulated as follow:

$$E=[\sum_{i=0}^{N-1}(C_iV_i+(-C_i)(1-V_i))-\log A]^2+\sum_{i=0}^{N-1}(V_i(1-V_i))$$

where $C_i = log\,(r_i)$ and $-C_i = log\,(1/r_i)$. It can be written in the standard form of Hopfield Model, where $T_{ij} = -8C_iC_j$ if $i \neq j$ and $T_{ii} = 0$, and $I_i = 4C_i(\log A + \sum_j C_j - C_i)$,

Experimental Results

In order to prove the the feasibility of Hopfield neural network in signal reconstruction from its FTM and one of its end points, the example presented in Reference[2] is given, N=20 and x(N-1)/x(0)=0.1332387. The root pairs (zeros of R(z)) are shown in Table I, along with the reconstructed signal.

We use PDP software of Boltzmann Machine named CS to solve the HNN. When initial temperature is high enough (about 20), the net almost converges to two sets of stable state. The results are the same as those in Reference[2].

The algorithms described above can be used to reconstruct signal which is either real or complex. In many practical uses, we try to retrieved real signal. The way to do this is to identify the pairs that hold conjugate roots and consider each pair of roots as one neuron. This will lead to decrease in computation time and network architecture(big saving!).

Conclusion and Feature Work

Suggested extensions include consideration of more efficient algorithm to solve this NP-complete problem, such as Mean Field Annealing (MFA) and Cauchy or Gaussian Machine. Also extension to 2-Dimensional signals.

Reference

[1] L.Xu, P.Yan and T. Chong,"Phase Information is richly contained in the End Point of a Finite Length Digital Signal," Proc. of the IEEE Intl. Symp. on Circuits and Systems, Philadelphia PA, Vol.3, pp. 851-854,1987.

[2] Haralambos Sahinoglou and Sergio D. Cabrera," On Phase Retrieval of Finite-length Sequences Using the Initial Time Sample," to be published on IEEE Transactions on Circuits and Systems.

[3] L. Xu, P. Yan and T. Chang,"Signal Reconstruction from its End Points and Fourier Transform Magnitude, Proc. of 1986 DSP workshop Sponsored by IEEE ASSP Society, sec. 4.5, Oct.20-22, 1986, Chatham, Massachusetts.

[4] J.Hopfield and D.Tank, Neural Computation of Decision in Optimization Problems, Biological Cybernetics, Vol.52,pp.141-152,1985.

[5] Henry Stark., Image Recovery: Theory and Application, Academic Press, Inc.,1987.

Table I Root Pairs Distribution with respect to Unit Circle of Ideal and Reconstructed Signals

Zeros No.	Magnitude of Zeros		Phase	Ideal	Case1	Case2
	Outside	Inside				
1	3.1046	0.3221	57.43	O	O	O
2	3.1046	0.3221	-57.43	O	O	O
3	2.3095	0.4330	-103.56	I	I	I
4	2.3095	0.4330	103.56	I	I	I
5	1.6984	0.5888	-47.82	I	I	O
6	1.6984	0.5888	47.82	I	I	O
7	1.6957	0.5897	-106.46	I	O	I
8	1.6957	0.5897	106.46	I	O	I
9	1.4465	0.6913	-160.09	I	I	I
10	1.4465	0.6913	160.09	I	I	I
11	1.3810	0.7241	59.80	O	I	I
12	1.3810	0.7241	-59.80	O	I	I
13	1.3160	0.7599	-19.00	I	I	I
14	1.3160	0.7599	19.00	I	I	I
15	1.1593	0.8626	115.89	O	I	I
16	1.1593	0.8626	-115.89	O	I	I
17	1.1396	0.8775	-8.890	I	I	I
18	1.1396	0.8775	8.890	I	I	I
19	1.1227	0.8907	0	O	I	I

Free-Form Deformable Surfaces for Object Modeling and Recognition[1]

M. Hebert, H. Delingette, K. Ikeuchi

The Robotics Institute,

School of Computer Science
Carnegie Mellon University
Pittsburgh PA 15213

Abstract

We present a technique for constructing shape representations from images using free-form deformable surfaces. We model an object as a closed surface that is deformed subject to attractive fields generated by input data points and features. Features affect the global shape of the surface while data points control its local shape. Our approach is used to segment objects even in cluttered or unstructured environment. The algorithm is general in that it makes few assumptions on the type of features, the nature of the data and the type of objects. We present results several applications: reconstruction of smooth isolated objects such as human faces, reconstruction of structured objects such as polyhedra, and segmentation of complex scenes with mutually occluding objects using multiple images. We describe the use of this algorithm in a mapping system for autonomous navigation.

1. Introduction

The recovery of object shape from 3D data is one of the key issues in vision. One could define this task as the segmentation of a large set of data points into shapes corresponding to objects in the scene. The shape representation should be general enough to handle a wide variety of scenes yet simple enough to be usable in other tasks such as recognition and manipulation. In other words, the shape representation should have enough parameters to describe the specificity of the shape but must have as few parameters as possible to be usable and to be robustly extracted from visual data. This is similar to the scale space problem, where one would like to find a description fine enough to capture the key details of the shape, but coarse enough to eliminate spurious data.

Several solutions have been proposed. Besl and Jain[2] built curvature-based object representations by classifying surfaces according to the sign of their principal curvatures. Pentland[16] presented a physically-based algorithm, to recover in a unique manner a model from a set of features and a set of vibration modes. Another approach is to use both features and range data in separate stages. In a first stage, features are grouped into hierarchical sets, according to geometric properties (symmetry, connexity, etc.) and in the second stage models are fit to the segmented parts. This idea of hierarchical representation was initiated by Marr and Nishihara[12] and pushed further by the seminal ACRONYM[5] vision system by using generalized cylinders. Pentland[14],[15]'s "Representation by parts" using deformed superquadrics, proved to have some successful results but encounters some limitations[7]. While the feature grouping requires some accurate feature extraction and high level reasoning, the fitting of superquadrics[14],[23] to range data has some unstable behavior, due to its non-linear nature, and is suitable primarily for smooth and simple shapes.

Those techniques attempt to represent all shapes by using a set of elementary shapes (superquadrics, generalized cylinders, parametric patches, etc.) that can be described by a few parameters. This is clearly beneficial from the point of view of object recognition which amounts to manipulating analytical equations of the elementary shapes. In practice, however, it restricts considerably the class of objects and scenes to which the techniques can be applied. More general representations could be obtained by adding degrees of freedom to the elementary shapes (e.g., adding tapering and bending to superquadrics). However, the non-linear fitting algorithms involved in the recovery of such shapes become rapidly computationally expensive and numerically unstable. Furthermore, most of those techniques assume that the observed scene is first segmented in regions corresponding to

---

[1] This research was supported in part by NASA under Grant NAGW 1175, and in part by DARPA through ARPA order No 4976. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, expressed or implied of those agencies

individual objects. The shape extraction algorithms are then applied to each object. However, accurately segmenting a scene is a hard problem in itself.

To address those problems, Terzopoulos and Witkin[11],[18],[19],[22] proposed the concept of deformable contours or surfaces that are subject to forces generated by image elements such as edgels. This work demonstrated that free-form shapes are powerful tools for shape representation. However, to perform correctly, the deformable surface had to be initialized close to the line of symmetry and it did not address the problem of segmenting the input image. Terzopoulos and Metaxas[20] in a recent paper combined the elementary shape and the deformable surface approaches to gain a local/global representation.

Our approach is to represent shapes by deformable surfaces as well. Given an observed scene, a surface is initialized in the vicinity of detected features. The surface is then deformed subject to forces generated by features and data points. The forces generated by data points control local shape, while forces generated by features control global shape. A smoothness energy is added to the deformation equations to take into account the fact that data and features may be sparse and noisy. Clearly, such free form surfaces can represent a large class of objects since few constraints are put on the resulting shape. Furthermore, the algorithm is robust enough so that it does not require a precise segmentation of the scene as an input.

2. Free-form deformable surfaces

The basic surface description is a free-form deformable surface subject to three types of forces, external, internal, and inertial forces that follow the laws of a dynamic mechanical system. The surface is therefore a function of time t: initialization corresponds to $t=0$ and final shape corresponds to $t=T_0$.

External forces are generated by input data points and input features. External forces must be carefully designed to obtain an optimal shape description that uses features to constrain global shape and data points to fine-tune local shape. Earlier work on surface reconstruction using splines required a complete mapping between data and surface points to be defined beforehand. This constraint restricts the geometry of the problem and requires the surface to be initially very close to the solution.

Internal forces ensures that the surfaces will not tear apart, fold onto itself, or exhibit high curvature points or sharp discontinuities in curvature. The other role of internal forces is to provide constraints in regions in which little or no data is available. This is similar to the regularization approach to surface reconstruction from sparse data[10],[19]. The standard way of defining internal forces is to define the corresponding energy as the integral over the surface of the magnitude of the first and second derivatives[1][10],[19],[24] which characterizes the surface smoothness. The relative importance of external and internal forces is a trade-off between accuracy and smoothness.

Inertial forces are generated by the motion of the surface as it evolves over time assuming that the surface has a non-zero mass. Inertial forces are necessary to model the deformable surface as a dynamic mechanical system.

2.1. General equations of motion.

The internal and external energies involved in the deformation of the surface are:

- **Smoothness energy $E_{\text{smoothness}}$**: The smoothness energy is related to the geometric property of the surface. The smoothness energy is internal in that it depends only on the shape of the surface in the vicinity of each point.
- **Feature energy E_{feature}**: The feature energy quantifies the interaction between the features and the surface; its magnitude is function of the distance between surface point and feature and of time t.
- **Data energy E_{data}**: The data energy quantifies the effect of data points on the surface.

To calculate the equilibrium position of the surface using mechanical systems theory, we need to introduce two additional inertial energy terms:

- **Kinetic energy T**: A mass μ is associated with each data point thus generating a kinetic energy term. Unlike other dynamic splines[11],[18],[22] our approach uses explicitly the kinematic energy to link the deformation of the surface with the minimization of energy.
- **Raleigh Dissipation energy D**: The dissipation term is added to simulate the exchange of energy between the dynamic surface and a virtual medium in which it evolves. This damping term is added to avoid cases in which the surface oscillates around an equilibrium position.

Following the equations of mechanics and the principle of least action[8], the surface reaches a stable equilibrium when the Lagrangian of the system of forces reaches a minimum. The Lagrangian of the system is: $L = T - D$

- $E_{\mathbf{smoothness}}$ - $E_{\mathbf{feature}}$- $E_{\mathbf{data}}$. A deformable surface is parametrized by two spatial coordinates (u,v) and by the time t. A surface point P has coordinates:

$$\vec{OP} = \vec{r}\,(u, v, t) = \begin{bmatrix} X\,(u, v, t) \\ Y\,(u, v, t) \\ Z\,(u, v, t) \end{bmatrix}$$

With these notations, using the calculus of variations on $L(u,v,t)$, one gets [18]:

$$\mu \cdot \frac{\partial^2}{\partial t}\vec{r}\,(u, v, t) = -k \cdot \frac{\partial}{\partial t}\vec{r}\,(u, v, t) + \vec{F}_{smoothness} + \vec{F}_{feature} + \vec{F}_{data}$$

where μ is the mass density of the surface and k is the damping factor.

2.2. Smoothness Energy

We use the bivariate generalized spline functionals of the first and second order as a smoothness measure[13],[19]. The first order is a measure of the distance discontinuities while the second order is a measure of the surface orientation discontinuities. Denoting partial derivatives by subscripts (e.g., $\vec{r}_u = \frac{\partial}{\partial u}\vec{r}\,(u, v, t)$), the energy is defined by:

$$E_{smoothness1} = \alpha_1 \cdot \int_0^{T_0} \left(\iint_{u\,v} (\|\vec{r}_u\|^2 + \|\vec{r}_v\|^2 \cdot du \cdot dv) \right) dt$$

$$E_{smoothness2} = \alpha_2 \cdot \int_0^{T_0} \left(\iint_{u\,v} (\|\vec{r}_{uu}\|^2 + 2\|\vec{r}_{uv}\|^2 + \|\vec{r}_{vv}\|^2 \cdot du \cdot dv) \right) dt$$

The constants α_1 and α_2 controls the relative smoothness of the surface. Let Δ_{uv} denotes the Laplacian operator with respect to (u,v). The corresponding force is therefore:

$$\vec{F}_{smoothness} = -\alpha_1 \cdot \begin{bmatrix} \Delta_{uv} X \\ \Delta_{uv} Y \\ \Delta_{uv} Z \end{bmatrix} + \alpha_2 \cdot \begin{bmatrix} \Delta_{uv}\Delta_{uv} X \\ \Delta_{uv}\Delta_{uv} Y \\ \Delta_{uv}\Delta_{uv} Z \end{bmatrix}$$

2.3. Feature Energy

Because every feature contributes to the global deformation of the surface, our approach is to link every feature to every point of the surface. Therefore, if $E^i_{feature}$ is the energy between a point on the surface and the feature number i, the total feature energy of the surface is:

$$E_{feature} = \sum_{i=0}^{n} \left[\int_0^{T_0} \left(\iint_{u\,v} (E^i_{feature} \cdot du \cdot dv) \right) dt \right]$$

Our algorithm satisfies two requirements. First, the initial surface can be “far” from the object so that only an approximate estimate of object location is required to initialize the iterations. Second, the points tend to accumulate near the features, thus guaranteeing that the major features of the objects are preserved.

To get both global and local deformations is difficult to achieve because the two types of deformation, feature and data, should be balanced. If only feature deformation were applied one would get a surface that connects the features with mostly planar surfaces in between because of the smoothness constraint (Figure 1(b)). On the contrary, in case of data deformation, one would get a surface that would faithfully follows the shape of the object only where the surface is close enough initially (Figure 1(c)).

Our solution is to change the relative influence of both types of deformations over time. Initially, the surface is mostly subject to feature forces, therefore moving toward its global shape; then the feature forces decrease and the surface is deformed locally so that it smoothly interpolates the shape of the object.To achieve this shift over time, the expression of $E^i_{feature}$ is defined as follows: let the distance between the feature and the point P on the surface be $D(Feature,P)$ and let a reference distance be $D_{\mathrm{ref}}(t)$at time t.

- if $D(Feature,P) > D_{\mathrm{ref}}(t)$ then the feature does not attract the point P:$E^i_{feature}$ = cste.
- if $D(Feature,P) < D_{\mathrm{ref}}(t)$ then the feature attracts the point P like a spring:$E^i_{feature}$ is proportional to

$[D(Feature,P)]^2$.

By setting $D_{ref}(t) = D_0^i \cdot (T_0 - t)$, as t increases, the surface has to get closer and closer to be attracted by the feature. Therefore the influence of the features becomes smaller during the deformation.

In order to avoid discontinuities, we use a formulation in which the energy function remains smooth when $D(Feature,P)$ is close to $D_{ref}(t)$. In the current implementation, we model features as 3-D line segments. We denote the midpoint of the segment by F_i, and its length by l_i. With these notations, the energy field generated by feature i, $E^i_{feature}$ (Figure 5(b)) is:

$$E^i_{feature} = U\left(\frac{l_i \| \vec{PF_i} \|}{D_0^i \cdot (T_0 - t)}\right)$$

where:

- $U(x)$ is a function that is quadratic if $x<1$, constant if $x>1$ and a cubic polynomial if x is close to 1 (Figure 2(a)). Therefore $U(x)$ is C^1 continuous, leading to an expression of the force that is C^0 continuous.
- $l_i \| \vec{PF_i} \|$ represents the distance between the point $P(u,v)$ of the surface and the feature segment of length l_i and middle point F_i. Other feature representations can be used in the same framework by replacing this term by the appropriate value $D(Feature,P)$. For example, for a point feature it would be the distance between surface point and feature point.
- D_0^i is the distance between the feature and the center of the surface at its initial position. If the surface is initialized as a sphere, it is the distance to the center of the sphere.

From the definition of feature energy, the force generated by the feature i at surface point P at time t is the opposite of the gradient of $E^i_{feature}$:

$$\vec{F}^i_{feature} = \frac{-l_i}{D_0 \cdot (T_0 - t) \cdot \| \vec{PF_i} \|} \cdot \frac{dU}{dx}\left(\frac{l_i \| \vec{PF_i} \|}{D^i_0 \cdot (T_0 - t)}\right) \cdot \vec{PF_i}$$

where dU/dx is the derivative of $U(x)$ (Figure 3(a)).

Figure 2(b) and 3(b) show the variation of the energy and force with respect to the time and $D(Feature,P)$; there is no singularity when t is close to T_0. By continuity, we can set $E^i_{feature}(t=T_0) = cste$ and $\vec{F}^i_{feature}(t=T_0) = 0$. Forces from each feature may be weighted to reflect the relative importance of different types of features.

2.4. Data Energy

Theoretically, a surface point is subject to forces from all the data points. However, for computational reasons we take into account only the closest data point. For every point P of the surface, the closest data point is denoted by C_{data}. Since data information is used only for local deformation, the corresponding force should decrease sharply with distance. Therefore, a gravity-type field where the energy decreases with the inverse of the distance is appropriate for this type of deformation. But to avoid the singularity when the distance is null, C_{data} acts like a spring when the point $P(u,v)$ is close to C_{data}. If E_{data} is the data energy of a point $P(u,v)$ of the surface:

$$E_{data} = W\left(\frac{\| \vec{PC}_{data} \|}{K}\right)$$

$W(x)$ is a function that is quadratic if $(x<1)$ and in $1/r$ if $(x>1)$ (Figure 4(a)). K is a normalizing constant that has the dimension of a distance. Intuitively, K represents the range of the influence of the data on the surface. In practice K is chosen small compared to the maximum expected object size. The corresponding force is:

$$\vec{F}_{data} = -\nabla E_{data} = \frac{\frac{dW}{dx}\left(\frac{\| PC_{data} \|}{K}\right)}{\| \vec{PC}_{data} \| K} \cdot \vec{PC}_{data}$$

where dW/dx is the derivative function of $W(x)$. Figure 4(b) shows the curve dW/dx. For $x>2$ the force is very small, which means that if $\| PC_{data} \| > 2.K$ the action of the data point is negligeable.

2.5. Implementation

We have assumed so far that our model is a continuous surface topologically equivalent to a sphere parameterized in (u,v,t). In practice, however, we can manipulate only discrete surfaces. This raises the problem of the parameterization of such surfaces and, in particular, the impossibility to map a sphere into a square in a uniform way. To avoid creating poles, we adopt the tessellated icosahedron as a structure. Each face of the icosahedron is subdivided to yield arbitrary resolution of the parameter space. The number of faces of the tessellation is $20N^2$, where N is the density of the subdivision. Typically, we use N=5 yielding a decomposition of the parameter space into 500 faces. We use the center of each triangle as a node, every node having therefore three neighbors. If we write $\vec{r}_t^{\,i}$ as the position of the node i at the time t, then the discrete version of the motion equation is:

$$\vec{r}_{t+1}^{\,i} = \vec{r}_t^{\,i} + (1-k) \cdot (\vec{r}_t^{\,i} - \vec{r}_{t-1}^{\,i}) + \vec{F}_{smoothness} + \vec{F}_{data} + \sum_{i=0}^{n} \vec{F}_{Feature}^{\,i}$$

The surface is initialized as a sphere at t=0 and is deformed by applying repeatedly the equation of motion at each node. Forces $\vec{F}_{data}$ and $\vec{F}_{feature}^{\,i}$ are computed at each node independently in a parallel manner. $\vec{F}_{smoothness}$ is computed by approximating the first and second derivatives of the surface by finite differences. The most expensive part of the algorithm is the computation of the closest data point C_{data} used in the computation of $\vec{F}_{data}$. It is theoretically in O(lm) where l is the number of data points and m the number on nodes. In practice, the search for the closest point is limited to a small subset of the m nodes by using the geometry of the sensor. The algorithm is otherwise linear in the number of features and the number of iterations. Due to its highly parallelizable nature, substantial speed-ups can be achieved.

Several parameters must be set to apply the motion equation. The parameter settings in the current implementation are:

- **Radius and center of initial sphere**: The determination of center and radius of the initial sphere depends on the nature of the data. The radius is related to the expected size of the object and the center position to the set of extracted features and a priori knowledge of the scene.
- **Number of iterations T_0**: Because T_0 is explicitly used for the computation of the feature force, it has an influence on the recovery of the object. The largest T_0 is and the smallest the deformation due to the feature is between two iterations and therefore the smoother the final shape is.
- **Smoothness coefficient α_1 and α_2**: The smoothness coefficients should be between 0 and 1 with $\alpha_1 > \alpha_2$. Actual "good" values have to be determined empirically. We use $\alpha_1 = 0.75$ and α_2=0.4 in our experiments.
- **Damping factor k**: The damping factor should be close to (but lower than) 1 to ensure smooth deformation of the surface over time and to avoid oscillations. We use k=0.9.
- **Normalizing factor K**: K depends on the environment. It is computed as one-fifth of the radius of the largest object expected in a scene for a given application.

3. Experimental Results

We report two sets of experimental results. In the first set, we took images of various scenes using a light-stripe range finder, and computed surface and object models from those images using the deformable surfaces algorithm. In the second set of experiments, we took series of range images from a mobile platform using a laser range finder. We developed a system for macthing objects between succesive frames and for building surface and object models using the deformable surface algorithm. The results show that the algorithm can be successfully applied to complex scenes without requiring a precise initial segmentation of the scene.

3.1. Experiments with Light-Stripe Range Data

For our experiments we use a commercial light-stripe range-finder that consists of a camera and a projector that projects patterns commanded by a LCD board[17]. Several sensors can be used at once to yield multiple views of a scene and thus demonstrating that the algorithm is completely independent of an image-centered reference frame. Our first experiment addresses the simplest case of a single image of an isolated object, a human face. The second experiment demonstrates the ability to deal with data from multiple sensor, and to handle arbitrary reference frames. The third experiment demonstrates how the algorithm can be used to segment a scene into individual object models.

3.1.1. Human Face

A human face is a good example of a complex object with parts of very distinct nature: the forehead and jaws areas are of little interest for face recognition, whereas eyes, nose, mouth and chin are the main characteristics of a face. Figure5(a)(b) show the intensity image and the corresponding range data of a human face. In a first experiment, we consider as features only the segments that surround the face (Figure 5(b)). We initialize the surface as a sphere so that it is roughly tangent to the face (Figure 5(c)). The resulting shape is displayed in two manners: Figure 6(a) is the triangular mesh of points while Figure5(e) is a shaded display obtained from ray-tracing. While the overall shape of the face was found, important facial features such as nose, mouth and eyebrow were smoothed.

To avoid this smoothing effect it is necessary to tag the eye, nose and mouth as being important features. The thresholding of the magnitude of an edge detector on the intensity image provides a way to extract those features (Figure 5(d)). The result of the final recovery is shown in Figure 5(f) and Figure 6(b): this time the nose, eyebrow and mouth are clearly visible and the left part of the nose was interpolated. This clearly demonstrates how the use of both features and data points enables a hierarchical description of the object.

3.1.2. Stacked Pebbles

The generality of our approach makes it particularly well-suited for unstructured environment in which there are few constraints on object shapes. The scene we use in this example consists in a set of pebbles that are lying on top of sand, using three range images from three different viewpoints (Figure 7(a)). Three different types of features (Figure 7(b)) are used: distance discontinuities, surface orientation discontinuities and shadow boundaries. Shadows are parts of the scene that are visible from the camera but are not illuminated by the projector. Final shaded display is shown in Figure 7(c).

This application to the segmentation of natural scenes is part of an operational system that automatically picks up pebbles in cluttered natural environment[6]. The discrete surface extracted from range images are fed to a program that computes the stable grasp positions for a three-finger gripper[9].

3.1.3. Stacked Cubes

To investigate further the ability of the algorithm to perform segmentation and surface reconstruction, we designed a simple scene with three overlapping cubes. Figure 8(a) is a view from one of the two range finders observing the scene: one cube is in balance between two adjacent cubes. The algorithm was applied successively to each of the three cubes. For each segmentation, a region around a cube is selected (square in Figure 8(a)), features are extracted and the center of the initial sphere is set et the center of this region. Figure 8(b)(c)(d) represent the three recovered cubes and Figure 8(e) the final reconstruction of the scene.

The algorithm is clearly able to automatically discard features and data that are not consistent with the surface. Among the three reconstructed shapes, the second seems to be the best (Figure 8(c)), mostly because it is not occluded by any other cube. On the other hand, the third cube (Figure 8(d)) has one face that is poorly recovered because there is no feature and no data corresponding to this face of the cube. Therefore all visible parts were correctly extracted.

3.2. Application to Map-Building Using a Laser Range Finder

An application of the deformable surface algorithms is to build models of objects and terrain in the environment of a mobile robot. Such a mapping system using superquadrics and generalized cylinders is described in [3]. Our approach is to use free-form surfaces to build models from partial or inaccurate range data of a large class of objects. To experiment with such a mapping system, we used the CMU Navlab[21], an autonomous vehicle designed for navigation on roads and in mild cross-country terrain. Navlab is equipped with cameras and a scanning laser range finder. The current Navlab system based on annotated maps is able to navigate through a previously mapped network of roads. It may take some action, such as stopping or switching sensor modality at specific landmarks. One of the main issues in making such a system robust is to be able to construct reliable maps of the environment. In particular, the locations of 3-D objects must be accurately recorded, and object models must be constructed. This involves identifying objects in individual images, tracking objects between images, and merging objects descriptions from individual images into composite models.

The mapping system is organized as follows: Features extracted from individual range and reflectance images are grouped in clusters corresponding to objects in the scene. A model of the surface of each object is computed from the features and data points within each cluster. Finally, objects are tracked from image to image to accumulate data and to remove spurious object detection.

The first step in building object models is to segment a range image into regions each of which corresponds to one object which may itself be segmented into smaller features. In practice, it may be hard to reliably extract such segmentations from individual images. To solve this problem, we generate an approximate representation of the scene by extracting simple features instead of attempting to generate a perfect segmentation, and we relegate the job of building accurate object representations to a higher-level shape representation module. The features are range and reflectance edges and near-vertical regions. Features are assembled into clusters. Each cluster is assumed to correspond to one object. Data points that are within a given distance of the cluster are included as well as features. This approach does not require an exact object/background separation or a precise segmentation.

Only incomplete models can be built from a single observation. As a result, object models should be refined as new data becomes available. Combining observations is critical to improve object localization, to remove spurious objects, to correct errors in the segmentation, and to accumulate data as the vehicle moves closer to the observed objects. Matching objects is straightforward in our case because we have only a few objects to match in each frame and because we have a reasonable estimate of the displacement between frames from INS or dead-reckoning so that the locations of the objects detected in one image can be easily predicted in the next image.

The problem of spurious objects is solved by calculating a confidence measure for each object. Once an object has been seen in one image, it should appear in subsequent images, as predicted by vehicle motion. If it appears as predicted, its confidence increases, otherwise it decreases. Objects with low confidence are discarded. Accurate object locations are obtained by updating the uncertainty on object location each time an object is observed in a new image. The uncertainties are combined using standard maximum likelihood techniques. Figure 9(a) shows a sequence of range images. The images are 64 rows by 256 columns with range coded on 8 bits. The images are separated by about 50 cm. The white lines connect the objects that are matched between images. Notice that some objects that are split in several clusters are merged into a single object in subsequent images. Also, the object at the left edge of the first two images is a spurious object that is not retained in the final map. Our approach to object representation and scene segmentation is ideally suited for model refinement. When a new range image is available, features are extracted and grouped into clusters as before. The new clusters are matched with clusters constructed from earlier observations using the tracking procedure described above. This is similar to the mechanism used in the TraX[3] except that we did not implement a full reasoning/explanation system. For each object, the surface fitting process is iterated using data and features from the previous images and from the corresponding cluster in the new image. This leads to a refined model of each object in the scene that takes into account the new data.

Figure 9(b) shows the result of applying the merging algorithms to the objects found in the sequence of images of Figure 9(a). The figure shows an overhead view of the terrain. Each object is modeled by a discrete surface with 500 vertices. The discrete surfaces are obtained by applying the deformable surface algorithm, starting with a triangulated sphere centered at the center of the corresponding cluster. Each discrete surface is iteratively deformed until it reaches a good compromise between smoothness and close fit to features and data points. Only a small portion of each object is visible in those images. This example shows that object models can be built automatically and refined by accumulating data and features from multiple images without requiring a precise initial segmentation of the image and in the presence of significant occlusions and noise.

4. Conclusion

We have designed and implemented an algorithm that constructs 3-D free-form deformable surfaces and is able to represent a large class of object without requiring perfect segmentation as an input. Based on the experiments, the algorithm has the following characteristics:

- **Physically based algorithm**: The dynamics of the deformation is modelled by the Lagrangian equations of mechanical systems.
- **Enhanced shape description:** Our algorithm enables to describe hierarchically the input data between two classes, data points and features, according to their influence on the overall shape of the object. This hierarchical description is embedded into the resulting model of the object.
- **Stability**: Because the algorithm uses both features and data, it is less sensitive to spurious features, noisy or missing data. Moreover, this stability enables to perform segmentation by discarding the features and data that is incompatible with the current shape of the surface.
- **Generality**: The algorithm makes few assumptions on data and observed objects. The only requirement is that some features can be extracted from input data and that the minimum and maximum sizes of the object expected in a typical scene are known. This allows us to deal with non-imaging sensors and merged multiple views.

The algorithms has some limitations as demonstrated in the experiments. First, it tends to smooth the sharpest features such as the corners of a polyhedron. This can be avoided by varying the smoothness parameters a_i in the vicinity of such a feature. Second, it generate an arbitrary surface model where no data is available. A mechanism should be added to identify which parts of the surface are actually valid, and to allow the integration of additional constraints that would force the surface to have specific properties, such as symmetry, in regions without data.

Future work concentrates on the use of the techniques presented in this paper in comprehensive robotics systems. We have already used deformable surfaces in a system for manipulating objects in natural environments using a three-finger gripper. We are currently using those techniques in a landmark mapping and recognition system for autonomous navigation[4].

References

[1] M. Berter, T. Poggio and V. Torre, *"Ill-Posed Problems in Early Vision*", M.I.T AI. Memo 924, 1987

[2] P. Besl and R. Jain, *"Segmentation Through Symbolic Surface Descriptions*", Proc. CVPR Miami, pp 77-85, 1986.

[3] B. Bolles and A. F. Bobick, *"Representation Space: An Approach to the Integration of Visual Information*", Proc. Image Understanding Workshop, 1989.

[4] M. Hebert, *"Building Object and Terrain Representations for an Autonomous Vehicle*", Proc. American Control Conference, June 1991.

[5] R.Brooks, *"Symbolic Reasoning Among 3D Models and 2D Images*". Artificial Intelligence Journal, 17:285-348, 1981.

[6] T. Choi, H. Delingette, M. Hebert and K. Ikeuchi, *"A Perception and Manipulation System for Collecting Rock Samples*", Proc. SOAR,1990.

[7] F. Ferrie, J. Lagarde and P. Whaite *"Darboux Frames, Snakes and Superquadrics: Geometry from the Bottom-Up"*, IEEE Workshop on Interpretation of 3D Scenes, Austin, 1989.

[8] C. Fox, *"An Introduction to the Calculus of Variations*", Dover Publications Inc., 1963.

[9] C. Francois, K. Ikeuchi and M. Hebert, *"A Three-Finger Gripper for Manipulation in Unstructured Environments*", Proc. IEEE Conf. on Robotics and Automation, 1991.

[10] K. Ikeuchi and B. Horn, *"Numerical Shape from Shading and Occluding Boundaries*", Art. Int. 17, pp141-184, 1981.

[11] M. Kass, A. Witkin and D. Terzopoulos, *"Snakes: Active Contour Models"*, Int. Journ. of Computer Vision, pp321-331,1988.

[12] D. Marr and K. Nishihara, *"Representation and Recognition of the Spatial Organization of Three Dimensional Shapes*", Proc. Roy. Soc. London, 200: pp269-294,1978.

[13] J. Meinguet, *"Multivariate interpolation at arbitrary points made simple"*, J. Applied Math. and Physics , 30, 292-304

[14] A. Pentland, *"Recognition by Parts*", Proc. ICCV London, pp 612-620, 1987.

[15] A. Pentland, *"Perceptual Organization and the Representation of Natural Form*", SRI International Tech. Rep. no 357, 1986.

[16] A. Pentland and S. Sclaroff, *"From Features to Solids*", M.I.T Media Lab Tech. Rep. No 135, 1990.

[17] K. Sato and S. Inokuchi, *"Range-Imaging System Utilizing Nematic Liquid Crystal Mask*", Proc. ICCV, pp 657-661, 1987

[18] D. Terzopoulos, A. Witkin and M. Kass, *"Symmetry-Seeking Models for 3D Object Reconstruction"*, Int. Journ. Comp. Vision, Vol 1, No 3 , pp211-221, Oct 1987.

[19] D. Terzopoulos, *"Computing Visible-Surface Representations*", M.I.T Art. Int. Memo 800, 1985.

[20] D. Terzopoulos, D. Metaxas, *"Dynamic 3D Models with Local and Global Deformations: Deformable Superquadrics*", Proc. ICCV, 606-615, 1990.

[21] C. E. Thorpe, *"Vision and Navigation, The Carnegie Mellon Navlab*", Kluwer Academic Publishers, 1990.

[22] A. Witkin, M. Kass, D. Terzopoulos, *"Physically Based Modeling for Vision and Graphics"*, Int. Journ. of Computer Vision, pp321-331, 1988.

[23] N.Yokoya, M. Kaneta and K. Yamamoto," *Recovery of Superquadrics Primitives from Range images by Simulated Annealing*", ETL Tech. Rep. TR-90-18, 1990.

[24] S. Zucker, *"The Organization of Curve Detection: Coarse Tangent Fields and Fine Spline Coverings*", Proc.

ICCV, pp 568-577,1988.

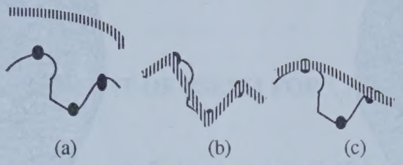


Figure 1: (a) Contour of an object with three features; (b) Final shape with feature attraction; (c) Final shape with data interaction only.

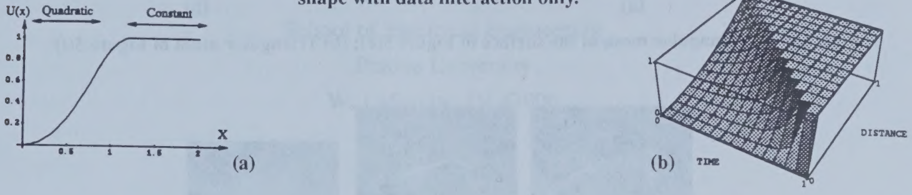


Figure 2: (a) $U(x)$; (b) $E_{feature} = f(t, D_{feature})$

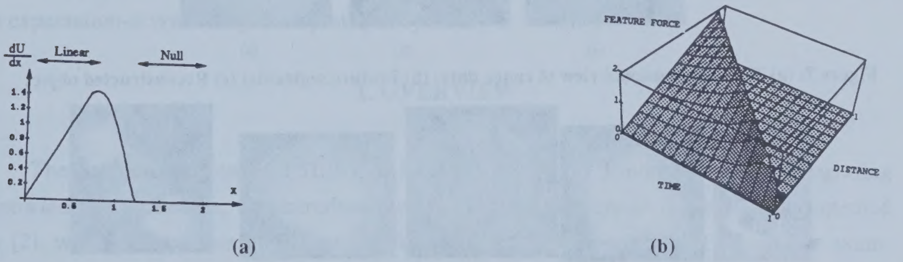


Figure 3: (a) dU/dx; (b) $F_{feature} = g(t, D_{feature})$

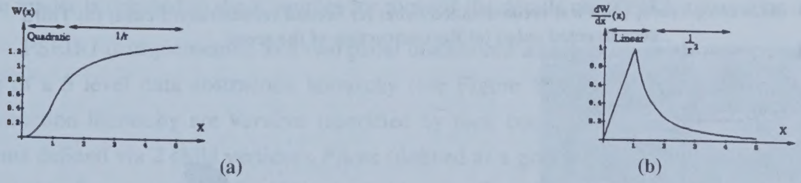


Figure 4: (a) $W(x)$; (b) dW/dx

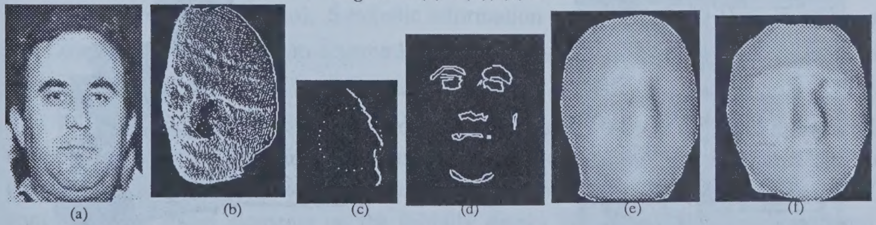
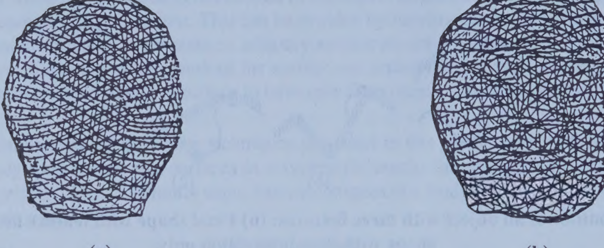
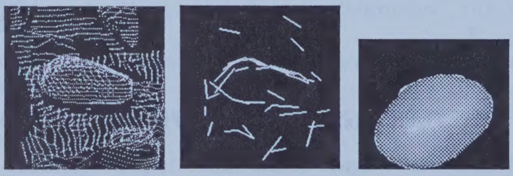


Figure 5: (a) Intensity image; (b) Corresponding range data; (c) Initial sphere; (d) Facial features; (e) Reconstruction of the face using only contour features; (f) Reconstruction of the face with facial features



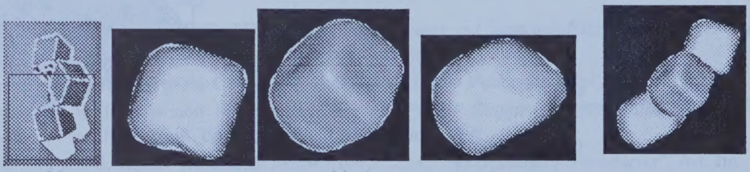
(a) (b)

Figure 6: (a) Triangular mesh of the surface of Figure 5(e); (b) Triangular mesh of Figure 5(f)



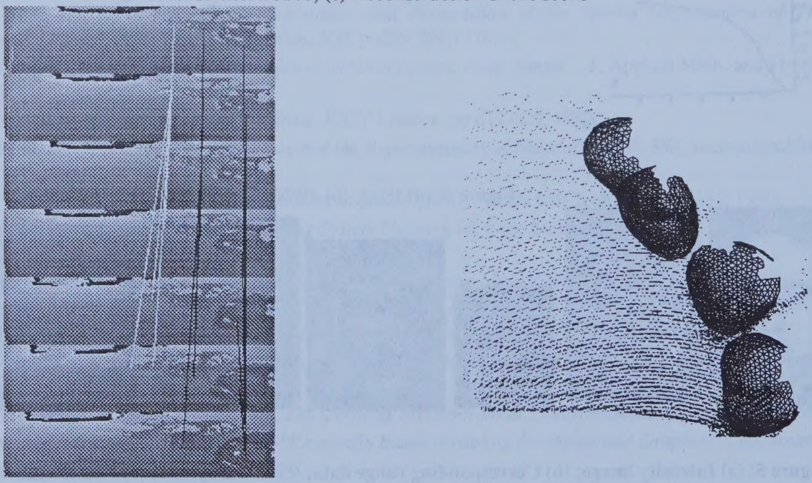
(a) (b) (c)

Figure 7: (a) Three dimensional view of range data; (b) Feature segments; (c) Reconstructed object



(a) (b) (c) (d) (e)

Figure 8: (a) Scene picture; (b) First reconstructed cube; (c) Second reconstructed cube; (d) Third reconstructed cube; (e) Reconstruction of the scene



(a) (b)

Figure 9: (a) Matching objects in a sequence of range images; (b) Overhead view of the objects models built using deformable surfaces

AN ASSESSMENT OF PSEIKI FOR CAD-BASED VISION

S. G. Blask and A. C. Kak
Robot Vision Lab
School of Electrical Engineering
Purdue University
W. Lafayette, IN 47906

ABSTRACT

This paper describes a series of experiments conducted to test the robustness of PSEIKI, an expectation-driven image interpretation system.

1. OVERVIEW

The original version of PSEIKI, a Production System Environment for Integrating Knowledge with Images, was introduced in [1]. The current version is formally documented by [2], while [3] contains a condensed informal description consisting of illustrative examples and an experimental application of PSEIKI to model-driven mobile robot navigation. The reader is referred to these sources for most of the details of PSEIKI's operation.

PSEIKI is implemented as a two panel blackboard architecture, with each panel consisting of a 5 level data abstraction hierarchy (see Figure 1). The levels of this VEFOS data abstraction hierarchy are *Vertices* (specified by their coordinates), *Edges* (straight line segments defined via 2 child vertices), *Faces* (defined as a group of edges), *Objects* (a collection of visible faces), and *Scenes* (the set of objects). Complete symbolic information from the Expected Scene Generator (typically a CAD modeler) is deposited on all 5 levels of the Model Panel of the blackboard. Symbolic information extracted from the observed image by the Low-Level Vision System (currently a region-based preprocessor) is deposited on the vertex through face levels of the Data Panel.

PSEIKI must find labels for elements initially posted on the Data Panel by determining the corresponding Model Panel elements. This is the task of the Labeler knowledge source (KS), which uses geometric constraints derived from the Model elements. PSEIKI must also construct Data Panel elements on the initially empty higher levels of the blackboard, for which the vision system does not provide data, by grouping together lower level Data Panel elements using the groupings present on the Model Panel as a guide. These elements are built by the Grouper KS, and then labeled by the Labeler KS. The Merger KS and Splitter

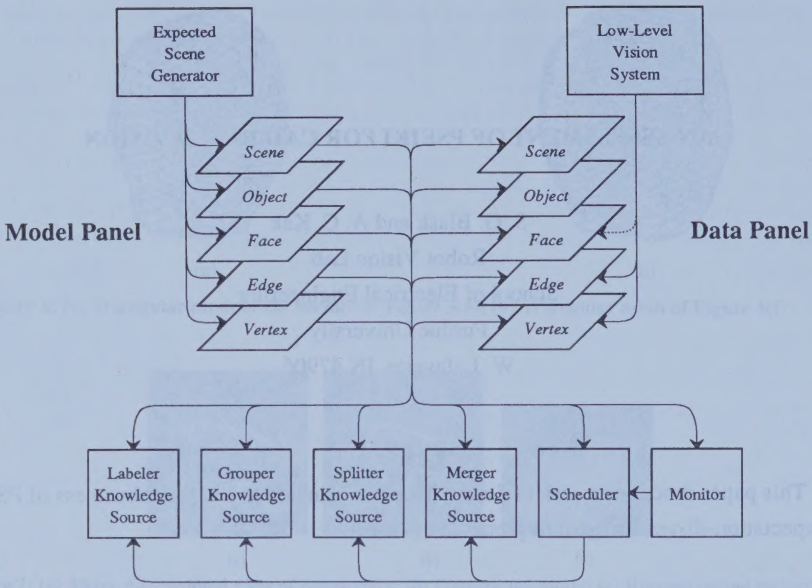

Expected Scene Generator
Low-Level Vision System
Model Panel
Data Panel
Scene
Object
Face
Edge
Vertex
Scene
Object
Face
Edge
Vertex
Labeler Knowledge Source
Grouper Knowledge Source
Splitter Knowledge Source
Merger Knowledge Source
Scheduler
Monitor

Figure 1 The configuration of PSEIKI's architecture.

KS attempt to correct grouping errors made by the Low-Level Vision System. Unlike the Grouper KS, the input and output elements of these two KSs reside on the same VEFOS level. The overall flow of control is determined by the Monitor acting in conjunction with the Scheduler. The Monitor is a daemon which detects when the preconditions of a KS are met, and builds a knowledge source activation record (KSAR) to trigger it. The Scheduler chooses which KSAR is to be fired during each cycle of the blackboard's operation. Once a KSAR has been fired, the KS it pertains to performs its action on the data associated with the KSAR and then returns control to the Scheduler. The system operates in 3 phases, *initializing* labels based on similarities to Model elements, *updating* beliefs based on geometric relationships between memebers of a group, and *propagating* evidence up the hierarchy.

2. MERITS

Given that the output of low-level processing is always corrupted by noise and other artifacts, it is imperative that a scene interpretation strategy use some kind of a belief system to keep track of confidence levels that can be associated with the various possible groupings extracted from the scene. While this need is universally recognized, by and large most attempts in this direction have been unyielding due to the peculiarities of the vision data. Unlike in purely symbolic systems, evidential reasoning for computer vision applications must be carried out against a background of geometrical calculations. Given that even scenes of only moderate complexity can result in a very large number of edges and/or regions, execution of geometric and belief calculations in some coherent framework is a formidable task.

The PSEIKI system tries to get around these difficulties by carrying out both the geometric and belief calculations in a data-driven opportunistic flow of control. What that means is that calculations, for the computation of beliefs or for geometric comparisons, need only be carried out in those parts of an image where the conditions are most optimum. Opportunistic flow of control is made possible by the OPS83 medium in which PSEIKI is programmed. OPS83, like its predecessor OPS5, maintains at all times a network of matches between the data and the preconditions of the rules, or more generally PSEIKI KSs. Any part of the system is immediately ready to fire the moment the data becomes available. PSEIKI also exploits the data abstraction hierarchy by performing a small number of computations at the higher levels in order to avoid many computations at the lower levels. Also, by first finding matches at the higher levels, PSEIKI is able to constrain the match process at the lower levels. PSEIKI performs label-driven belief accumulation using Barnett's formulas [4] as a starting point, and then through iterative updating yields the final belief even more efficiently.

3. EVALUATION

IHSE, an **I**nteractive **H**ierarchical **S**cene **E**ditor [5], was a critical tool used in the evaluation of PSEIKI's abilities. Its interactive data creation and manipulation capabilities were used to generate the root data from which the symbolic input data for both the Data and Model panels of PSEIKI were derived. Its ability to model camera projection, location, and orientation parameters allowed us to quantitatively study the effects of varying the viewpoint. Its input parser was used as a filter to remove redundancies from PSEIKI output. And finally, its 3D interactive graphics display capabilities made it useful for examining the resulting image interpretation produced by PSEIKI.

The first series of experiments used to evaluate PSEIKI consisted of simple template images. The purpose of these experiments was to determine the sensitivity of PSEIKI to locational variations of the observed scene (modeling incorrect estimation of the camera position in a mobile robot navigation context) while keeping the expected scene fixed.

The squares and blob templates shown in Figures 2-3 were created using IHSE. The sizes of the templates were constrained such that they fit within the bounding rectangle (−3, −3, 0) to (3, 3, 0). Projections of the templates onto the 2D image plane defined by the camera parameters *viewport size* 512*x*512 pixels, *eye point* at (0, 0, 18), *look direction* initially (0, 0, −1), *up direction* initially (0, 1, 0), *field of view* of 45°, and *perspective* projection with *focal length* equal to 0.050, were output in symbolic form for use as input to both the Model and the Data panels of PSEIKI. The viewpoint for the expected scene was fixed for all runs, i.e., the symbolic information deposited on the Model Panel was always the same aforementioned view of the template centered in the image. Observed scene projections for the Data Panel were created with one of the following "distortions" present at a time: horizontal image shift, vertical image shift, tilt angle, pan angle, and swing angle.

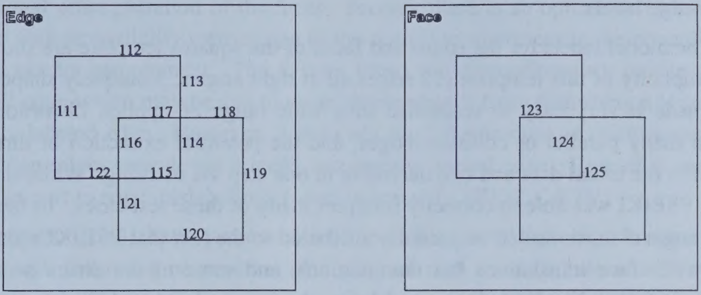
Horizontal and vertical shifts of the screen coordinate data were varied from −180 to +180 pixels by 10 pixel increments. Tilt and pan angles were varied from −18° to +18° in single degree increments. Tilting and panning by 1° roughly corresponds to vertical and horizontal shifting by 10 pixels, respectively, with the addition of perspective projection distortions. Swing angle (about the look direction, i.e., viewing vector) was varied such that approximately 36 samples at equal intervals of rotation were produced before the object became realigned with itself due to its symmetry, i.e., before the model rotated in the positive direction and the model rotated in the negative direction became isomorphic. These were the only distortions present in the image data; edges were not broken or missing, and closed face groupings were provided exactly as they occurred in the model.

In order to simulate region-based preprocessing, only elements from the vertex, edge, and face levels were deposited on the Data Panel. Symbolic information for the single object and single scene present in the templates was retained for the expectation Model Panel. PSEIKI was run on this data with a maximum misregistration of *Dmax*=150 pixels and 3 initial competing hypotheses, and the results were filtered through IHSE to remove redundant output.

Table 1: Template Processing Results

Amount	squares					blob				
↓	**H**	**V**	**T**	**P**	**S**	**H**	**V**	**T**	**P**	**S**
−18	A	A	A	A	D	P	P	P	P	M
−16	A	A	A	A	A	P	P	P	P	P
−14	A		A	A	D	H	I	P	P	P
−12	A			A	D	H	I	K	H	P
−10				A	D	H	I	K	H	P
−8				C	E	I	I	K	H	P
−6				C		I	I	K	L	P
−4				C		I	I	K	L	P
−2				C		I	I	K	L	N
0						I	I	I	L	N
+2				C	C	I	I	I	L	N
+4			C		C	I	I	I	L	P
+6			C			I	I	I	L	P
+8			C	B	F	I	I	I	J	P
+10				B	D	J	I	I	J	P
+12	B		C	B	D	J	I	I	J	P
+14	B		B	B	D	J	I	P	J	O
+16	B	B	B	B	G	J	P	P	P	M
+18	B	B	B	B	G	J	P	P	P	M
units→	10	10	1	1	5	10	10	1	1	10

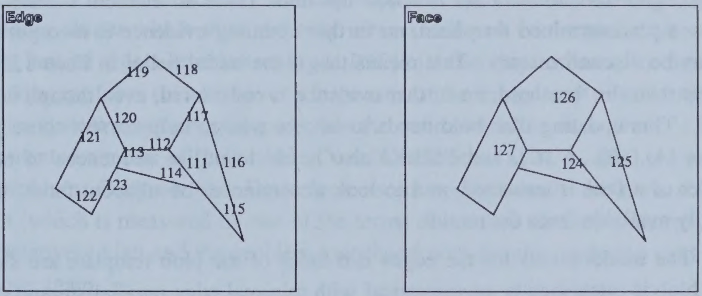
Table 1 summarizes the processing results produced by PSEIKI for the templates. It has been condensed by eliminating the odd numbered test cases, but still faithfully illustrates PSEIKI performance. To determine the amount of **H**orizontal or **V**ertical image shift or the **T**ilt, **P**an, or **S**wing angle distortion, multiply the constant in the *Amount* column by the constant in the *units* row to yield the quantity in pixels or degrees for the corresponding table entry. The shaded table entries correspond to distortions which cause the template to be clipped by the image boundary. If the table entry is blank, the interpretation produced by PSEIKI was totally correct. Otherwise, see the note corresponding to the table entry in Figures 2-3 for a description of the incorrect labels assigned by PSEIKI. If faces are mislabeled, of course their child edges are mislabeled. However, the converse is not true, i.e., the labels assigned to a set of edges grouped into a correctly labeled face might be a permutation of the


Edge
112
113
111
117
118
116
114
122
115
119
121
120
Face
123
124
125

Applicable notes from Table 1:

A. Face 125 mislabeled as 123.
B. Face 123 mislabeled as 125.
C. Edge 120 mislabeled as 115.
D. All edges mislabeled, although faces are correct.
E. Edges 114,116 mislabeled as 117,115.
F. Edges 115,117,120 mislabeled as 116,114,115.
G. Faces 123,125 mislabeled as 125,123.

Figure 2 Model labels for the edges and faces of the squares template.


Edge
119
118
120
117
121
112
113
111
116
114
123
122
115
Face
126
127
124
125

Applicable notes from Table 1:

H. Edge 113 merged into edge 12, face 125 mislabeled as 127.
I. Edges 113,114,116,121 mislabeled as edges 123,112,111,123.
J. Face 127 mislabeled as face 125.
K. Edges 111,113-116,121 mislabeled as edges 115,123,112,111,111,123.
L. Edges (111,)113,114,116,121 mislabeled as edges (115),123,112,111,123, and edge 111 & 115 sometimes merged.
M. Faces 125,127 mislabeled as faces 127,125.
N. Some edges mislabeled, although all faces are correct.
O. Faces 125,126,127 mislabeled as faces 127,125,126.
P. Various faces merged.

Figure 3 Model labels for the edges and faces of the blob template.

proper edge labels.

The model labels for the edges and faces of the squares template are shown in Figure 2. The simplicity of this template (12 edges all at right angles, 3 uniquely shaped faces) should have made it very easy to recognize in a wide range of applied distortions. Difficulties include many parallel or collinear edges, and the potential extraction of three squares, one present in the initial data and two derivable in one step via merging. While it did make some errors, PSEIKI was able to correctly interpret many of these test cases. In fact, the errors for swing angle distortions can be partially attributed to the fact that PSEIKI's similarity metrics account for face translations but not rotations, and some of the errors occurring near the extremes of the other distortions were due to the template being shifted beyond the specified *Dmax*. However, several cases of notes *(A)* and *(B)* are manifestations of the same problem for the face level that note *(C)* exemplifies for the edge level. The root of this problem is the fact that PSEIKI uses proximity to determine initial labels on the observed Data Panel, and then updates using only pairwise consistency checks among sibling data elements rather than longer sequences of adjacent labels from the Model Panel.

Recall that PSEIKI assigns initial labels to orphan elements deposited on the Data Panel based on their proximity to expected Model element locations. For note *(A)*, the misregistration of the observed data is such that the squares template appears sufficiently far above or to the left of the expectation for the Labeler to assign Face 125 an initial label of 123. This problem is compounded by the fact that PSEIKI attempts to keep processing time down by limiting the amount of updating evidence that is generated. It does this by not allowing all elements to have their belief function updated. Once an element's belief in its label rises above a predetermined threshold, no further updating evidence is incorporated, even though it may be disconfirmatory. This means that if the initial belief in Face 125's label of 123 is higher than the threshold, no further evidence is considered, even though it might correct the label. This updating threshold needs to be removed to help correct some problems causing errors *(A), (B), & (C)*, and PSEIKI also needs to utilize sequences of edge labels as the border of a face is traversed, and to look at sequences of adjacent faces, both of which are readily available from the model.

The model labels for the edges and faces of the blob template are shown in Figure 3. The blob is intentionally asymmetrical with minimal edge parallelism and collinearity, a mix of acute, right, and obtuse angles, and face groupings of various shapes and sizes (in terms of both area and number of edges). Despite the unique properties of each face, PSEIKI was not able to produce a single totally correct interpretation of a scene containing this template, not even when the model and data were in perfect alignment (no distortion). The results show several examples of groupings which have the proper content but the wrong permutation of labels (e.g., notes *(I), (K), (L), & (N)*), which calls for sequences of labels to be checked before propagating evidence up the hierarchy.

This template also exemplifies several problems associated with PSEIKI's current set of metrics used to measure (dis)similarity or (in)compatibility between elements. First of all, when faces are aligned before the computation of the edge level (dis)similarity metrics and during the computation of the face level (dis)similarity and (in)compatibility metrics, the transformation used to implement the alignment takes into account translation only, with no regard for rotational misregistration of the faces. Second, there is an optimistic bias present in the edge level (in)compatibility metrics due to the *scale()* term present in the noncollinearity measure but not its complement. The system limits the disconfirmatory evidence generated by a small edge which may be due to noise, preventing it from disturbing a large, supposedly correctly labeled edge. However, it does not limit the amount of confirmatory evidence such an edge might provide for a large, incorrectly labeled edge. Both of these problems contribute in part to edge mislabelings found in notes *(I), (K), (L), & (N).*

Previous papers [1,2,3] have presented mobile robot navigation experiments in the domains of sidewalk following or traversal of a planned path through indoor hallways. In these experiments, the camera of the robot was aimed low in order to limit the amount of visible data in the scene. We introduce another hallway experiment in which the camera has a more common, less restrictive viewpoint which provides a more complex and challenging scene. Figure 4 shows the observed scene and the output of the region-based Low-Level Vision System for the robot positioned 6 meters from the door at the end of a corridor. Figure 5 shows the faces of the Model Panel as rendered by IHSE, and the resulting interpretation produced by PSEIKI which was run with a maximum misregistration of *Dmax*=150 pixels and 3 initial competing hypotheses.

The first major problem for this scene is that PSEIKI does not really do true object level reasoning. All faces in the scene are grouped into a single object by the Expected Scene Generator, and there is no provision for moving objects or parts of objects which move relative to each other. Therefore only models of objects possessing a degree of permanence are provided to the Expected Scene Generator, with the hope that the excluded moving objects will appear as random clutter which obscures only some small part of the scene. Another consequence of the lack of object level reasoning is PSEIKI's inability to handle occlusions intelligently. Nonadjacent faces scattered over a large area of the image may be assigned the same label, with the system never checking whether the entire set taken as a whole is consistent with a fragmented version of their label. Data faces labeled as K, L, & Q are prime examples of this problem. Because these Data elements are approximately the same size as the Model element (which is measured by one of the terms of the face similarity metric), the initial beliefs are relatively high and the problem associated with pruning updating evidence comes into play once again.

In the current system all model elements on a given blackboard level have the same priority, and the sheer number of them in this typical scene bogs the system down. 111582 OPS83 rules were fired for 239 KS invocations and 1211 triggered KSARs, taking 1.5 hours of processing time on a SUN3 workstation. If PSEIKI's Expected Scene Generator could provide anticipated contrast information via lighting and material property models common to many of today's CAD systems, the Scheduler could use it to narrow the focus of its search more efficiently. The system could look for predicted high contrast elements first, which would also be the elements most reliably extracted by the Low-Level Vision System, thereby reducing the amount of data considered during a cursory match, and then build the interpretation out from there.

Like the faces of the blob template, the mopboard Data element which got labeled as Model element P is an example of an error which could have been prevented if some form of

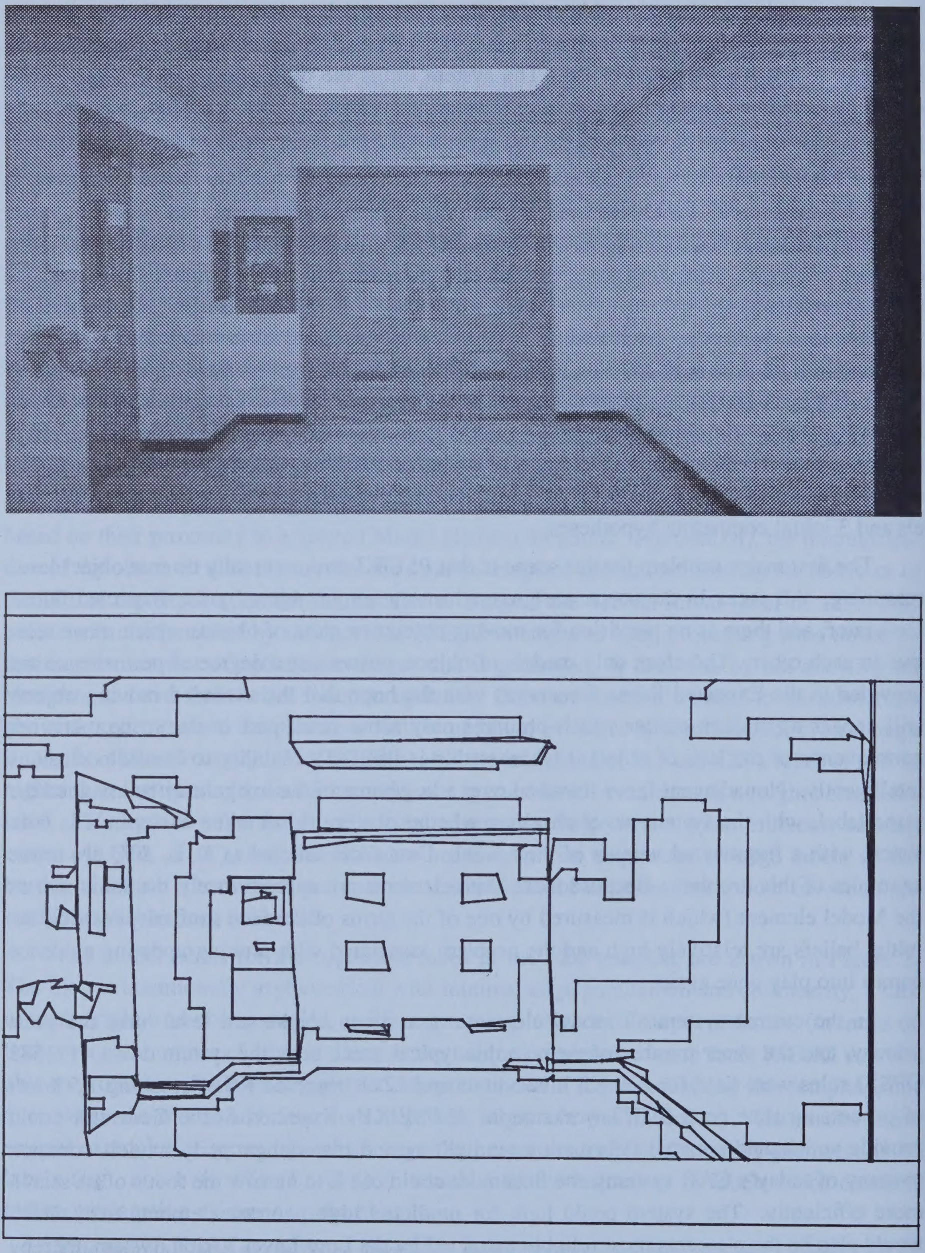


Figure 4 Digitized hallway image (above) with regions extracted by the Low-Level Vision System (below).

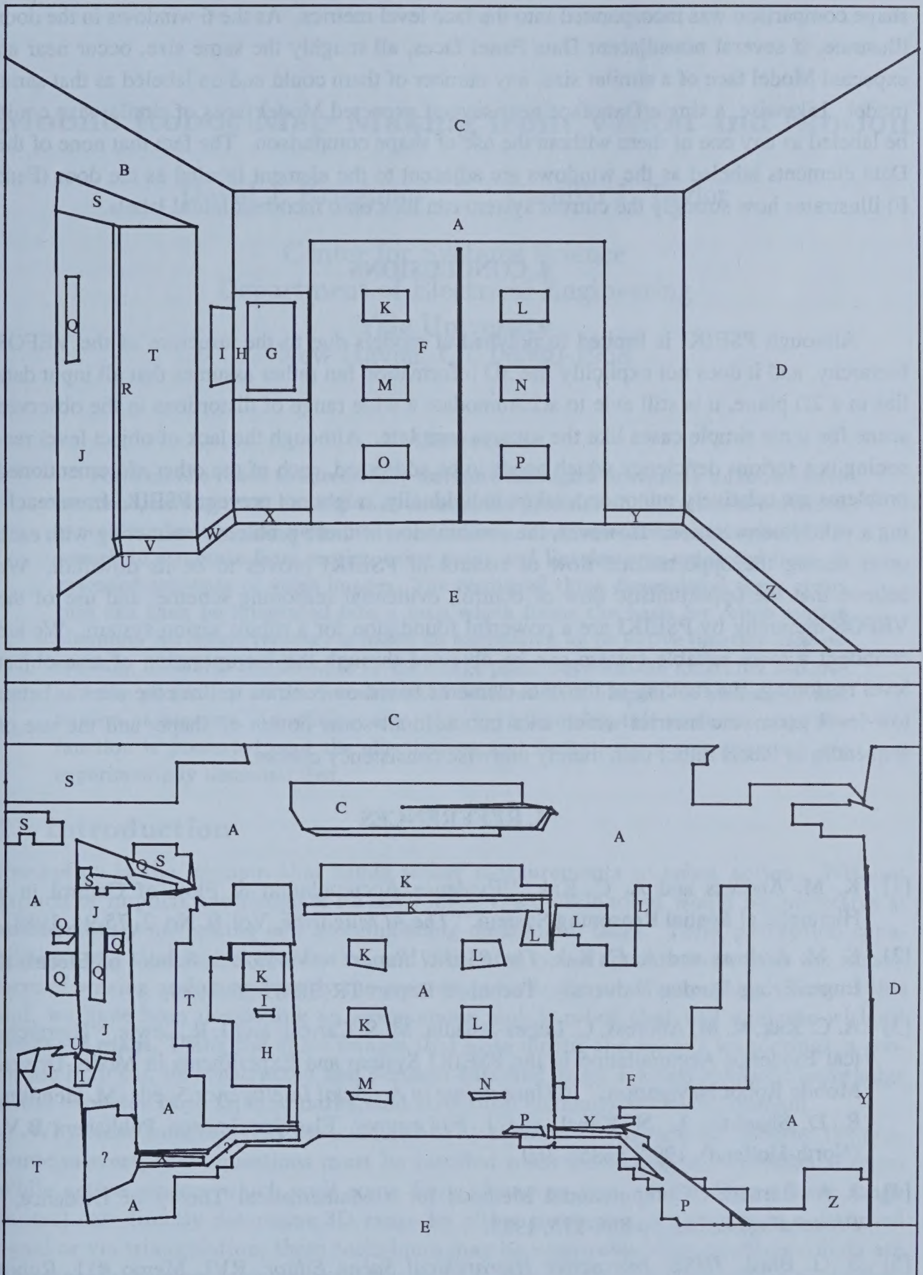

C
B
S
A
T
R
Q
I
H
G
K
L
F
M
N
D
J
O
P
X
Y
W
V
U
E
Z
C
S
C
S
A
A
Q
S
S
K
K
A
J
L
L
L
Q
Q
K
L
Q
Q
K
A
T
I
D
K
L
J
U
H
J
I
M
N
F
Y
A
P
Y
?
?
T
A
P
Z
E

Figure 5 Model faces from Expected Scene Generator (above), and resulting PSEIKI correspondences (below).

shape comparison was incorporated into the face level metrics. As the 6 windows in the door illustrate, if several nonadjacent Data Panel faces, all roughly the same size, occur near an expected Model face of a similar size, any number of them could end up labeled as that same model. Likewise, a single Data face near several expected Model faces of similar size could be labeled as any one of them without the use of shape comparison. The fact that none of the Data elements labeled as the windows are adjacent to the element labeled as the door (Face F) illustrates how strongly the current system can lock onto incorrect initial labels.

4. CONCLUSIONS

Although PSEIKI is limited to polyhedral models due to the structure of the VEFOS hierarchy, and it does not explicitly use 3D information but rather assumes that all input data lies in a 2D plane, it is still able to accommodate a wide range of distortions in the observed scene for some simple cases like the squares template. Although the lack of object level reasoning is a serious deficiency which needs to be addressed, each of the other aforementioned problems are relatively minor and, taken individually, might not prevent PSEIKI from reaching a valid interpretation. However, the combination of these problems interacting with each other during the opportunistic flow of control of PSEIKI proves to be its downfall. We believe that the opportunistic flow of control, evidential reasoning scheme, and use of the VEFOS hierarchy by PSEIKI are a powerful foundation for a robust vision system. We are confident a more reliable system can be obtained through the incorporation of true object level reasoning, the ranking of the data elements based on contrast to limit the search, better low-level geometric metrics which take into account some notion of shape, and the use of sequences of labels rather than merely pairwise consistency checks.

5. REFERENCES

[1] K. M. Andress and A. C. Kak, ''Evidence Accumulation & Flow of Control in a Hierarchical Spatial Reasoning System,'' *The AI Magazine,* Vol. 9, No. 2, 75-94, 1988.

[2] K. M. Andress and A. C. Kak, *The PSEIKI Report -- Version 3,* School of Electrical Engineering, Purdue University, Technical Report TR-EE 89-35, 1989.

[3] A. C. Kak, K. M. Andress, C. Lopez-Abadia, M. S. Carroll, and J. R. Lewis, ''Hierarchical Evidence Accumulation in the PSEIKI System and Experiments in Model-Driven Mobile Robot Navigation,'' in *Uncertainty in Artificial Intelligence 5,* eds. M. Henrion, R. D. Shachter, L. N. Kanal, and J. F. Lemmer, Elsevier Science Publishers B.V. (North-Holland), 1990, pp353-369.

[4] J. A. Barnett, ''Computational Methods for a Mathematical Theory of Evidence,'' *Proceedings IJCAI,* pp 868-875, 1981.

[5] S. G. Blask, *IHSE: Interactive Hierarchical Scene Editor,* RVL Memo #11, Robot Vision Lab, School of Electrical Engineering, Purdue University, 1991.

Mobile Robot Map Making from Vision and Motion

David J. Kriegman   Camillo J. Taylor

Center for Systems Science
Department of Electrical Engineering
Yale University
New Haven, CT 06520-1968

Abstract

For a mobile robot to successfully navigate through a previously unknown environment, sensing must be used to determine the presence and location of obstacles, generic landmarks, and perhaps its destination. This paper focuses on a batch approach to structure from motion using point and line features extracted from an extended sequence of video images. The recovered three dimensional scene structure can then be integrated into a map which forms the basis for robot motion planning. Structure recovery is based on minimizing an objective function that essentially measures the distance in the image plane between the measured features and the projection of the reconstructed features with respect to structural and motion parameters. An efficient algorithm for minimizing this nonlinear objective function is presented, and the effectiveness and limitations of this approach are experimentally demonstrated.

1 Introduction

Perception is the linchpin that binds sensor measurements to robot action. Without explicit or implicit knowledge of its environment, a mobile robot would be ineffective at autonomously navigating and accomplishing meaningful tasks. Thus, perceptual capabilities are a vital aspect of robotics, and computer vision appears to provide the richest form of sensing and consequently the greatest promise for robot perception. Toward this end, we have been developing an autonomous mobile robot that will navigate without prestored maps. Using the video images that arise during the robot's wanderings, a geometric map can be constructed and perhaps enriched using *a priori* symbolic knowledge of the expected objects, landmarks, and structural elements within its domain.

Why video images? Why one camera? Why structure from motion? Besides the academic interest, these questions must be justified when developing a perception scheme. While active sensors which emit some form of energy (e.g. SONAR and laser range finders) can directly determine 3D range by either measuring the energy in a returned signal or via triangulation, these techniques may be unsuitable when multiple robots are interacting and when environmental factors (e.g. specular surfaces) cannot be controlled. Alternatively, passive sensing with video cameras can provide similar depth information, and more importantly surface markings, which do not necessarily correspond to depth discontinuities, can be located; consider finding a closed door in a hallway. Certainly a great deal of information is available in a single monocular image. However, three dimensional (3D) scene geometry cannot be determined unless models of the observed objects are known *a priori.* Two prominent alternatives for constructing a 3D representation from intensity images are stereo vision and structure from motion. The necessary assumptions for using other image cues to recover scene geometry such as texture, shading and contour shape are presently too restrictive to be used for mobile robotics. While stereo vision holds great promise, it has a number of drawbacks. The field of view is inherently limited with two or more cameras, and solving the correspondence problem can often be difficult and expensive. Alternatively, the presented approach to structure from motion effectively increases the field of view by integrating measurements observed over a subset of the extended image sequence. By tracking image features at near frame rates, the correspondence problem is bypassed. Additionally, sensor cost is reduced since only a single camera is required.

There are two major categories of approaches to structure from motion: those based on image velocities and those based on tracking image features (tokens). In the first category, either the optic flow at all image points is determined from two or more images [1, 11], and a dense range map and ego-motion are calculated from retinal velocities [3] or the so called direct methods can be applied. These methods bypass explicit flow calculations, and directly determine ego-motion from the image sequence; range is subsequently computed [10, 17]. In the approach considered in this paper, features are detected in monocular images and tracked through an image sequence [7]; structure and motion parameters are then found simultaneously. When surfaces are textured, optic flow or direct approaches are advantageous because it is difficult to detect features in a single image (e.g. consider aerial views of treetops). Most algorithms based on image flow are restricted to two frames which effectively limits the baseline used in structure recover; however, recent approaches for more accurately estimating flow over multiple frames appear promising [2]. In indoor scenes, features are relatively sparse, and token tracking is more appropriate. Additionally, feature correspondences can be maintained over long baselines, potentially leading to more accurate range estimates.

There is a wide literature on the problem of structure from motion using feature correspondences which include different image formation models (perspective vs. orthographic projection), different feature types (regions, points, lines), varying numbers of images and features, and finally different types of algorithms. For a mobile robot, important structural elements of the world can have large spatial extent, so a perspective projection model must be used; it should be noted however that the results of recent approaches, which assume either distant objects and orthographic projection [23] or shallow structures and local affine projection [19], are both impressive and computationally advantageous. In this paper, we consider both point and line features, though we are strongly biased toward linear features since they are very prevalent in indoor scenes and are very accurately located in images. While most previous research has been concerned with point correspondences, some notable approaches based on line correspondences include [8, 13, 16, 20]. Early approaches to structure from motion only considered the minimum number of images and measurements [24]; these were shortly followed by two frame algorithms that take advantage of additional features [14]. More recent multi-frame approaches are either batch algorithms, where all data is considered simultaneously [22, 23] or incremental algorithms which typically employ a Kalman filter to recursively update parameter estimates [8, 15].

In this paper, a new batch approach is presented for recovering observer motion and scene structure from the straight line segments and points found in a sequence of three or more images. The recovered structural elements are the 3D points and infinite straight lines that are in best agreement with the observed 2D points and finite straight line segments for the estimated camera positions. More precisely, the problem is formulated for lines as minimizing the total squared distance in the image plane between the observed line segments and the perspective projection of the reconstructed infinite 3D straight lines with respect to the parameters representing the 3D lines and observer motion. A similar notion can be applied to points. While the projection of the recovered line onto an image plane has infinite extent, only the portion that is near the measured line segment contributes to the error measure. Note that the proposed algorithm endeavors to derive the best possible estimates by directly minimizing the non-linear objective function and avoids the linear approximations made by extended Kalman filter approaches; it can, therefore, be expected to yield more accurate results. When the line segments are vertical or the points are coplanar and the robot motion is also planar, the presented algorithm degenerates to the least squares approach to recovering planar point locations and motion presented in [22]. While a generic nonlinear least squares technique such as the Levenberg-Marquardt algorithm [18] could be employed to find a local minimum, the objective function can be partitioned into a number of smaller problems, leading to an efficient optimization scheme.

After discussing the structure from motion problem and the presented algorithm in more detail in secs. 2, 3, we present some experimental results in section 4. In section 5, we consider a parallel computational architecture for implementing this approach on a mobile robot and finally conclude in sec. 6 with some future directions.

2 Problem Formulation

The image formation process can be modeled as a function $\mathcal{F}$ which takes the position of the camera $\mathbf{q}$ and the position of a three-dimensional feature and returns a representation for the corresponding feature in the image. In this paper, we consider features to be either points $\mathbf{p}$ or straight lines $\mathbf{l}$, and the camera to be modelled by pinhole perspective with unity focal length. Consider the case where there are m images of a static scene containing n_p points and n_l straight line segments, let $\tilde{\mathbf{p}}_{ij}$ be the measurement of the projection of point i in image j; the image of a line is similarly denoted as $\tilde{\mathbf{l}}_{ij}$. Let $Dist_p()$ and $Dist_l()$ be functions that measures the disparity between the observed image features and the expected image features given by the projection functions $\mathcal{F}_p$ and $\mathcal{F}_l$. An objective function $\mathcal{O}$ can now be defined:

$$\mathcal{O} = \frac{1}{\sigma_p} \sum_{j=1}^{m} \sum_{i=1}^{n_p} Dist_p(\mathcal{F}_p(\mathbf{p}_i, \mathbf{q}_j), \tilde{\mathbf{p}}_{ij}) + \frac{1}{\sigma_l} \sum_{j=1}^{m} \sum_{i=1}^{n_l} Dist_l(\mathcal{F}_l(\mathbf{l}_i, \mathbf{q}_j), \tilde{\mathbf{l}}_{ij}) \tag{1}$$

where σ_p and σ_l are constants that are related to the accuracy of the measured points and lines. Here we consider the accuracy to be identical for each feature type, however the signal to noise ratio (related to image contrast, line length, etc.) of each measured feature could be incorporated. Also note that $\mathcal{O}$ can be trivially modified so that observing each feature in every image is unnecessary.

This objective function measures the total disparity between the expected image measurements and the actual image measurements. Our aim then, is to find a choice of parameters, $\mathbf{p}_i$, $\mathbf{l}_i$ and $\mathbf{q}_j$, that is most consistent with the actual image measurements by minimizing $\mathcal{O}$ with respect to these parameters. In the remainder of this section, the elements of (1) are discussed in more detail.

The camera positions $\mathbf{q}_j$ are parameterized in the usual manner; a vector $\mathbf{t}_j \in \mathbb{R}^3$ and a rotation matrix $\mathbf{R}_j \in SO(3)$ are used to specify the position and orientation of the camera with respect to the base frame of reference. Note that the rotation matrix R_j can be rewritten as $exp\{J(\mathbf{k}_j)\}$, where $\mathbf{k}_j \in \mathbb{R}^3$ and J is the skew symmetric operator [6]. This means that the camera position $\mathbf{q}_j$ is uniquely specified by the vectors $\mathbf{t}_j \in \mathbb{R}^3$ and $\mathbf{k}_j \in \mathbb{R}^3$, a total of six independent parameters. We now proceed to define the representation, projection function and disparity function for points and lines.

2.1 Points

The recovered three dimensional points are simply represented by their coordinates, $\mathbf{p} \in \mathbb{R}^3$ in the base frame. To define the projection function, the coordinates of $\mathbf{p}$ are written in the camera frame in the usual way according to the rotation $\mathbf{R}$ and translation $\mathbf{t}$ that relate the camera frame to the base frame.

$$ {}^c\mathbf{p} = \mathbf{R}^T(\mathbf{p} - \mathbf{t}) \tag{2} $$

Considering the camera to have unit focal length and the optical axis to lie along the z-axis of the camera frame, the image coordinates of ${}^c\mathbf{p} = ({}^cx, {}^cy, {}^cz)$ are given by the usual pinhole projection equations $(x', y') = ({}^cx/{}^cz, {}^cy/{}^cz)$. Together these two equations define the projection function $\mathcal{F}_p$. The disparity function is simply the squared Euclidean distance between the measured point $\tilde{\mathbf{p}}$ and the projection of the reconstructed point (x', y'). Note that (x', y') is function of both $\mathbf{p}$ and $\mathbf{q}$.

2.2 Lines

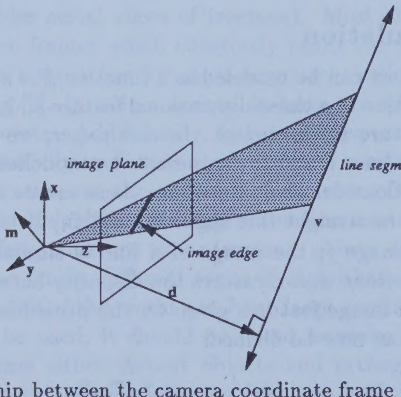


Figure 1. The relationship between the camera coordinate frame and an infinite straight line.

The algorithm presented in this paper is designed to recover the infinite straight lines that support the three-dimensional line segments in the scene, rather than the line segments themselves. This makes the algorithm more robust to problems that can occur when parts of a segment are occluded. Figure 1 shows the relationship between an infinite straight line and the camera frame. The unit vector $\hat{\mathbf{v}}$ indicates the direction of the line, the vector $\mathbf{d}$ designates the point on the line that is closest to the origin, while the unit vector $\hat{\mathbf{m}}$ is the normal to the plane through the origin that includes the line.

Note that since the three vectors, $\hat{\mathbf{m}}$, $\hat{\mathbf{v}}$ and $\mathbf{d}$ are mutually orthogonal, it is possible to construct an orthonormal matrix $L \in SO(3)$ such that $L = [\hat{\mathbf{m}}|\hat{\mathbf{v}}|\hat{\mathbf{d}}]$ where $\hat{\mathbf{d}} = \mathbf{d}/||\mathbf{d}||$. This observation suggests that the straight line can be represented by an orthonormal matrix $L \in SO(3)$ and the scalar value $||\mathbf{d}||$. Since an orthonormal matrix can be parameterized by a vector $\mathbf{r} \in \mathbb{R}^3$, this representation involves a total of four independent parameters.

Before arriving at a projection function $\mathcal{F}_l$ for straight lines, the parameters of the line must be given with respect to the camera frame in terms of $\mathbf{R}$ and $\mathbf{t}$, and we will denote the transformed parameters by $({}^c\hat{\mathbf{v}}, {}^c\hat{\mathbf{m}}, {}^c\mathbf{d})$. In particular, they are given by:

$$\begin{aligned} {}^c\hat{\mathbf{v}} &= \mathbf{R}\hat{\mathbf{v}} \\ {}^c\mathbf{d} &= \mathbf{R}(\mathbf{d} - \mathbf{t} - (\mathbf{t} \cdot \hat{\mathbf{v}})\hat{\mathbf{v}}) \\ {}^c\mathbf{m} &= {}^c\hat{\mathbf{v}} \times {}^c\mathbf{d} \end{aligned} \tag{3}$$

Note that (3) is not a simple rigid body transformation, but instead it relates the representation of the line to the camera frame. The projection function $\mathcal{F}_l$ for a line onto the image is simply the intersection of the image plane $z = 1$ with the plane defined by the camera origin and ${}^c\mathbf{m}$, and is given by $m_x x + m_y y + m_z = 0$ where ${}^c\hat{\mathbf{m}} = {}^c\mathbf{m}/\|{}^c\mathbf{m}\| = (m_x, m_y, m_z)^T$.

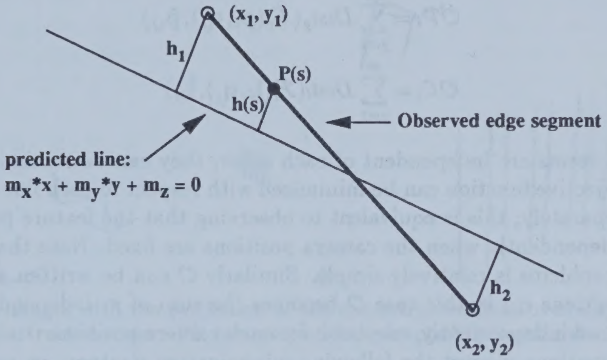


Figure 2. *Dist*$_l$: The distance between the observed and predicted segments.

Finally, we define the disparity function $dist_l()$ as the total perpendicular distance between the measured line segment and the projection of the infinite reconstructed line. Given the end points of the measured line segment (x_1, y_1) and (x_2, y_2), an equation for the observed segment can be parameterized by a scalar value $s \in [0, l]$ where l is its length. Similarly, the normal distance $h(s)$ from a point on the measured segment to the reconstructed line can be defined as illustrated in figure 2. The total distance between the observed edge segment and the predicted edge can then be expressed as:

$$Dist_l = \int_0^l h^2(s)ds = \frac{l}{3}(h_1^2 + h_1 h_2 + h_2^2) \tag{4}$$

where h_1 and h_2 are the normal distances of the end points. Finally, it should be reiterated that the measured end points do not necessarily correspond to the end points of the three-dimensional line segments.

3 Recovery Algorithm

The recovery algorithm takes a set of noisy images and a set of estimated camera positions provided by odometry and proceeds to determine the scene structure and camera positions by iteratively minimizing the objective function presented in section 2. Given m images, n_p point features, and n_l straight line features, this objective function involves $(6m+3n_p+4n_l)$ independent parameters, quite a challenging problem for general purpose non-linear minimization algorithms. However, we can take advantage of the structure of the objective function to devise a fast, effective minimization strategy. Specifically, the optimization problem can be decomposed into a number of smaller, simpler problems by dividing the parameters of interest into two mutually exclusive sets: the structural parameters and the camera positions.

For example, when the objective function is rewritten in terms of the structural parameters, it becomes the sum of $n_p + n_l$ independent terms where n_p terms are denoted $\mathcal{OP}_i$, one for each point, and n_l terms are denoted $\mathcal{OL}_i$, one for each line.

$$\mathcal{O} = \sum_{i=1}^{n_p} \mathcal{OP}_i + \sum_{i=1}^{n_l} \mathcal{OL}_i \tag{5}$$

where

$$\mathcal{OP}_i = \sum_{j=1}^{m} Dist_p(\mathcal{F}_p(\mathbf{p}_i, \mathbf{q}_j), \tilde{\mathbf{p}}_{ij}) \tag{6}$$

$$\mathcal{OL}_i = \sum_{j=1}^{m} Dist_l(\mathcal{F}_l(\mathbf{l}_i, \mathbf{q}_j), \tilde{\mathbf{l}}_{ij}) \tag{7}$$

Since these terms are independent of each other, they can be minimized in parallel. That is, the objective function can be minimized with respect to each line and each point in the scene separately; this is equivalent to observing that the feature positions can be determined independently when the camera positions are fixed. Note that each of these minimization problems is relatively simple. Similarly $\mathcal{O}$ can be written as a function of the camera positions $\mathbf{q}_j$; in this case $\mathcal{O}$ becomes the sum of m independent terms that can be minimized independently, one term for each camera position.

These observations suggest the following minimization strategy; on each iteration of the algorithm the objective function is minimized with respect to each set of parameters in turn. Each of these minimization steps can be decomposed into a number of simpler minimization operations which can be carried out in parallel. These operations will either decrease the overall value of the objective function or leave its value unchanged, this ensures that the algorithm will eventually converge to a local minima. Initial estimates of $\mathbf{q}_i, \mathbf{p}_i$ and $\mathbf{l}_i$ are required for the minimization. Odometry provides estimates of $\mathbf{q}_i$; estimates of $\mathbf{p}_i$ and $\mathbf{l}_i$ can be obtained by considering the multieyed stereo problem where the relative camera positions are given by $\mathbf{q}_i$.

4 Experimental Results

Experiments have been performed using both simulated and real motion sequences. The simulation experiments provide a quantitative evaluation the algorithm's performance by systematically varying a number of factors such as noise level, number of images, number of edges, feature configurations, and robot motion. To qualitatively demonstrate successful reconstruction, the algorithm has been applied to a few real image sequences.

Here, we summarize the results of the simulation experiments; for more details, please see [21, 22]. In the first experiment, a scene composed of line segments randomly located within a 1m cube was observed from four locations 2m away from the segments; the mean 3D distance between the actual segments and the reconstructed lines was 3.86cm, and the mean error in the reconstructed line direction $\mathbf{v}$ was 1.5°. The mean error was 2.59cm in camera location and 1.39° in camera orientation. In this case, image noise was uniformly distributed within ±1 pixel, and odometry errors of ±5° and ±5cm were

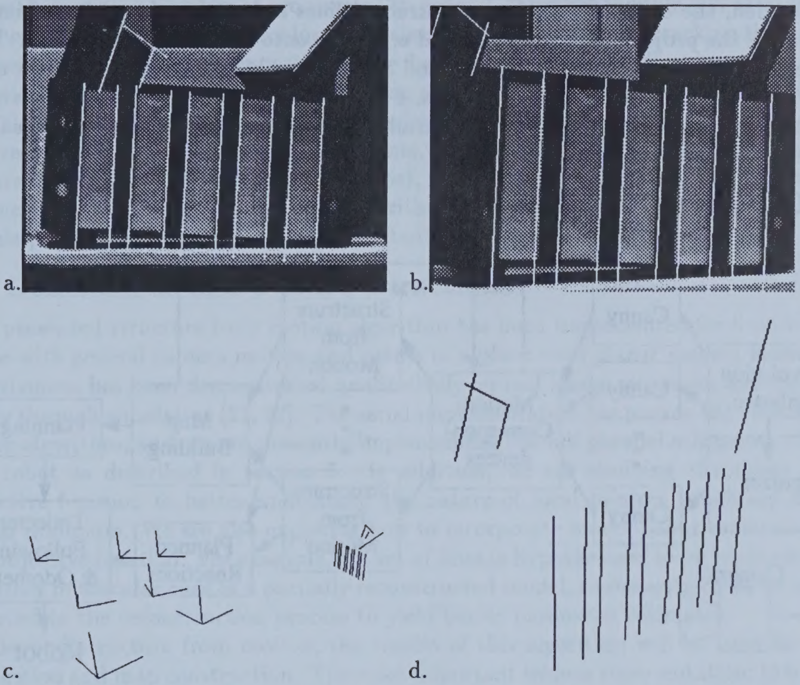


Figure 3. Two images with the projection of the reconstructed lines overlaid, and two other views of the reconstructed scene with camera positions denoted by coordinate frames.

introduced. In a second experiment, we explored the effect of observing more features on egomotion estimates; as expected, additional features impose more constraints on robot location, and consequently the pose is estimated more accurately. Specifically, for ten point features lying in a plane with planar robot motion, the RMS error was 2.21cm as opposed to 1.62cm for fifty features. We also explored the effect of increasing the number of images on the reconstruction error. For planar motion and thirty points features, the RMS error decreased from 19.2cm for three images to 10.9cm for nine images. Note that for any structure from motion algorithm "your mileage may vary," as reconstruction accuracy is highly dependent on scene structure and robot motion. The second and third experiments corresponded to a configuration similar to a robot moving down a corridor where observed points lie near the optical axis and the reconstruction accuracy deteriorates.

The presented algorithm has been applied to a small number of image sequences that were obtained using a CCD camera with an 25mm lens mounted on a TRC Labmate mobile robot. The intrinsic camera parameters (focal length, pixel aspect ratio, camera center and quadratic distortion) were determined using a least squares technique. Image edges were found using an implementation of Canny's edge detector [4] and motion correspondences were specified manually. Estimates of robot location were determined from odometry.

Figures 3.a,b show two images in a sequence of 10 images with 14 corresponding lines. The line segments drawn on the images represent the projection of the reconstructed 3D lines onto the image planes for the estimated camera locations. For the sake of presentation, the endpoints of the reconstructed lines are crudely found by taking the extrema of the projection of the measured segments onto the reconstructed line. Thus, we make no claims about their accuracy and are principally interested in the error of the supporting line and the measured segment. Figures 3.c,d show two different views of the reconstructed segments. Note that the parallel lines on the front of the storage cabinet are nearly coplanar in the reconstruction.

5 Computer Architecture

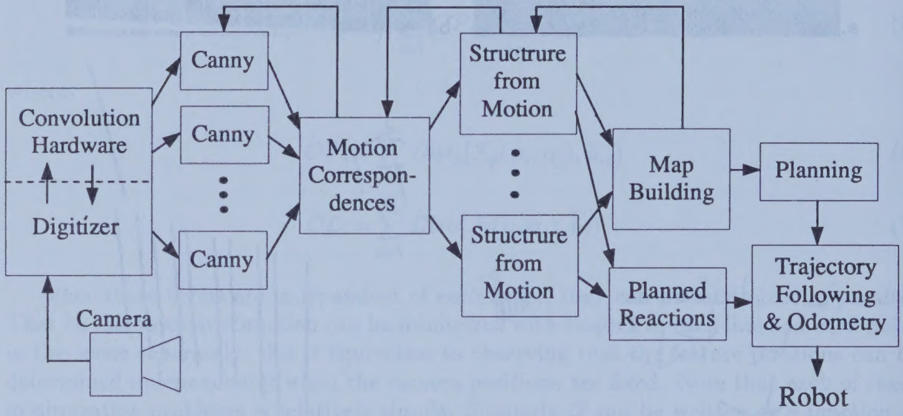


Figure 4. The pipelined/parallel architecture used for image processing, tracking, and structure from motion. Each box represent at least one processing node.

Currently, the structure from motion (SFM) algorithm has only been implemented on a serial UNIX workstation. However, we are presently constructing a pipelined/parallel implementation on a network of INMOS transputers residing on our mobile robot. Figure 4 shows a block diagram of the architecture where each box represents at least one processing node, and the arrows indicate information flow. Each node contains a transputer (25Mhz, 12 MIPS, 1.5 MFLOPS), 1–4 Mbyte of memory, and four 20 Mbit per second bidirectional communication links.

We hope to simplify determining motion correspondences by tracking features at near frame rates as the robot moves. To maintain this high rate, processing bandwidth is more important than latency. Thus, we have arrived at a pipelined architecture. While frame n is digitized, at least four processors are extracting the Canny edges from frame $n-1$. The digitizer contains hardware that can convolve an image with a 6×7 kernel in real time. The extracted straight lines are passed to another processor which determines the correspondence between the previously observed features and the newly extracted ones. While features are extracted from frame $n-1$, the correspondences of frame $n-2$ are determined. Feature tracking may also include a predictive component using either a 2D dynamical model of edge motion in the image plane [7] or a 3D dynamical model that would account for robot motion and the reconstructed feature location produced by the SFM processors. Additionally, the feature detectors might only be applied where correspondences are expected.

Simultaneously, the SFM processors are updating the structural and motion estimates in frame $n-3$ using the presented algorithm. Recall that the algorithm can be executed in parallel across a number of processors. In general, the SFM module will already have good estimates for many feature locations since they will have been tracking the features for some time. Thus, we expect that the batch minimization will only require a few iterations to converge. Finally, the resulting structural information will be passed on to modules (processors) that are responsible for map maintenance, "long term" planning and responsive action to time critical events. Finally, a processor is devoted to executing trajectories (including emergency stops), monitoring robot status, and maintaining odometry estimates used by the SFM algorithm. We have not considered robot planning in this paper, and leave this all too important issue open to speculation.

6 Conclusions and Future Directions

The presented structure from motion algorithm has been implemented for lines in three space with general camera motion and points in a plane with planar camera motion. Its effectiveness has been demonstrated qualitatively on real image sequences and quantitatively through simulation [21, 22]. The serial implementation has proven the effectiveness of the algorithm, and we are presently implementing the full parallel scheme on the mobile robot as described in section 5. In addition, we are studying the shape of the objective function to better understand the nature of local minima which are not the global minimum. We are also exploring how to incorporate model-based constraints into the objective function. For example, if a set of lines is hypothesized to be coplanar based on either monocular cues or a partially reconstructed model, coplanarity could be strictly enforced in the reconstruction process to yield better parameter estimates.

Beyond structure from motion, the results of this algorithm will be used for robot navigation and map construction. The most important issue is representation; this choice affects the accessibility of information, the ease of incorporating new information, the consequences of measurement uncertainty and the impact of errors in observation/map correspondences [9]. It has often been argued that because of odometry error, maps should not represent feature location in some absolute coordinate system but should instead, maintain relational information. Unfortunately, most previous work that has accounted for odometry error has either relied on an absolute coordinate system with some indication of measurement error (e.g. Gaussian distributions) [12], a collection of local models [5], or even eschewing representation altogether. Instead we propose to use a map based on relationships between features that are invariant under isometries (i.e. distances and angles) along with their estimated variances. By maintaining a sparse subset of these relationships, the map size will grow linearly with respect to the number of features; contrast this with a map based on absolute coordinates where a full covariance matrix grows by the square of the number of features. Some important questions to be resolved are: Which features should be represented? What parameterizations are appropriate? How can the information be easily accessed? How difficult is it to construct a Cartesian map (which is useful for motion planning) from the relational map? How is new information incorporated? How can symbolic information be extracted? Can tasks be planned directly from this representation?

Acknowledgments:

We would like to thank P. Anandan for many useful discussions and insights. Support for this work has been provided in part by a gift from the INMOS division of SGS-Thomson.

References

[1] P. Anandan. A computational framework and an algorithm for the measurement of structure from motion. *International Journal of Computer Vision*, 2:283–310, 1989.

[2] M. Black and P. Anandan. Robust dynamic motion estimation over time. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, pages 296–302, June 1991.

[3] A. Bruss and B. Horn. Passive navigation. *Comp. Vision, Graphics, and Image Proc.*, 21(1), 1983.

[4] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Machine Intell.*, pages 679–98, Nov. 1986.

[5] J. Crowley. Navigation for an intelligent mobile robot. *IEEE Journal of Robotics and Automation*, pages 31–41, Mar. 1985.

[6] M. Curtis. *Matrix Groups.* Springer-Verlag, 1979.

[7] R. Deriche and O. Faugeras. Tracking line segments. *Image and Vision Computing*, pages 261–270, 1991.

[8] O. D. Faugeras, F. Lustaman, and G. Toscani. Motion and structure from point and line matches. In *ICCV*, pages 25–33, June 87.

[9] G. Hager. *Task Directed Sensor Fusion and Planning.* Kluwer, Boston, 1990.

[10] K. J. Hanna. Direct multi-resolution estimation of ego-motion and structure from motion. In *IEEE Workshop on Visual Motion*, pages 156–161, 1991.

[11] B. Horn and B. Schunck. Determing optical flow. *Artificial Intelligence*, 17:185–204, 1981.

[12] D. J. Kriegman, E. Triendl, and T. O. Binford. Stereo vision and navigation in buildings for mobile robots. *IEEE Trans. on Robotics and Automation*, 5(6):792–803, Dec. 1989.

[13] Y. Liu and T. S. Huang. A linear algorithm for motion estimation using straight line correspondences. In *Int. Conf. Pattern Recognition*, 1988.

[14] H. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.

[15] L. Matthies, R. Szeliski, and T. Kanade. Kalman filter-based algorithms for estimating depth from image sequences. *International Journal of Computer Vision*, 3:293–312, 1989.

[16] A. Mitiche, S. Seida, and J. Aggarwal. Determining position and displacement in space from images. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, 1985.

[17] S. Negahdaripour. *Direct Passive Navigation.* PhD thesis, MIT, 1986.

[18] W. Press, B. Flannery, S. Teukolsky, and W. Vetterling. *Numerical Recipes in C.* Cambridge University Press, 1988.

[19] H. S. Sawhney and A. R. Hanson. Identification and 3D description of "shallow" environmental structure from a sequence of images. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, 1991.

[20] M. E. Spetsakis and J. Aloimonos. Structure from motion using line correspondences. *International Journal of Computer Vision*, 4(3):171–184, June 1990.

[21] C. Taylor and D. Kriegman. Structure and motion from line segments in multiple images. In *IEEE Int. Conf. on Robotics and Automation*, 1992. Submitted. Paper available from the authors.

[22] C. Taylor, D. Kriegman, and P. Anandan. Structure and motion in two dimensions from multiple images: A least squares approach. In *IEEE Workshop on Visual Motion*, 1991.

[23] C. Tomasi and T. Kanade. Shape and motion without depth. In *IUW*, 1990.

[24] S. Ullman. *The Interpretation of Visual Motion.* The MIT Press, Cambridge, MA, 1979.

THREE-DIMENSIONAL RIGID-BODY POSITIONING USING CAMERA-SPACE CONTROL

U.A. KORDE W.Z. CHEN S.B. SKAAR

Dept. of Aerospace & Mechanical Engineering,
University of Notre Dame, Notre Dame, IN 46667

ABSTRACT

This paper presents results of experiments undertaken to test the vision-based method of camera-space manipulation for a three-dimensional rigid-body mating task. The task objective is to place one object (the "entering piece", grasped by the end-effector) at a certain position and orientation with reference to the cartesian coordinate system fixed to another object (the "receiving piece") which itself may be placed at an uncertainly known location. In this work, the sensitivity of the success rate of this method to various operating factors is studied experimentally using a large 6-axis manipulator. Among the factors examined are relative orientation and number of cameras, weighting of measurements (involving the entering piece) used in the estimation procedure, camera model, etc. Finally, an analytical study is carried out to understand the influence of relative orientation of cameras and their number on the terminal positioning precision expected with the method.

1. INTRODUCTION

The use of vision allows a manufacturing robot to perform a task at a work-piece whose position and orientation in cartesian space may be uncertainly known and/or subject to external disturbances. In such an operation, information from the images acquired by the vision sensor (for instance, a CCTV or CCD camera) is used to guide the end-effector to its desired position and orientation. However, because this information is 2-dimensional while the physical space is 3-dimensional, implementing vision-based position control is generally not straightforward. Therefore, in most applications it is necessary to find a means of relating the 3-dimensional position and orientation of the end-effector with the information gathered by the camera. Such a relation would be a function of the robot kinematics, camera position and orientation relative to the cartesian reference frame attached to the robot base (relative to which the kinematics may be formulated), and camera internal parameters such as effective focal length, digitization scaling factors, lens distortion effects, etc. (See Tsai[1]). In the literature, the problem of vision-guided positioning of the grasped object has been approached in a number of different ways.

One approach involves 'calibration' of the camera (or a pair of stereo cameras) prior to the maneuver by observing a number of accurately known physical space points (calibration points) and estimating the parameters in a model that relates the cartesian coordinates with their image-plane coordinates. This approach was followed, for instance, by Ishii[2] *et al* in the development of a robot teaching method, and by Abidi & Gonzalez[3] in rigid-body mating tasks. A version of the same calibration approach can be found in the work employing structured lighting in conjunction with a pre-established range map. This technique was used, for instance, by Mochizuki[4] *et al* in a task related to printed-circuit-board assembly, and by Agapakis[5] in an arc-welding application. The parameters evaluated in the calibration procedure are sometimes used directly to evaluate the joint coordinates necessary for achieving a particular physical space configuration.

Alternatively, visual feedback may be employed, where the difference between the desired and current image-plane locations of points/features may be used to generate actuating signals for each joint based on the above calibration (as in Shirai & Inoue[6], and Tani[7] *et al*; in tasks requiring insertion of rectangular blocks into rectangular openings). Visual feedback has been demonstrated in tasks where the correct position and orientation of the grasped object may be obtained by exercising only 3 degrees of freedom. Implementations of such feedback in tasks requiring 6 degrees of freedom have been proposed, (See Weiss[8] *et al*, Feddema & Lee[9]), but to the best of our knowledge, no experimental results have been reported.

There may be some essential limitations to the methods based on pre-maneuver calibration. Firstly, they require the exact kinematic model, and hence it is likely that the elaborate procedure of kinematic calibration (See Hollerbach[10]) would be necessary. Secondly, the coordinate transformation between the reference frame mounted on the robot base and the camera reference frame must be known precisely. Again, this may be difficult to measure with a great degree of precision. The use of stereo cameras requires further that the exact relative position and orientation of the two cameras be available, and presents the need for identification of conjugate points (See Horn[11]). Further, in the event of any changes in the camera position, orientation, or internal parameters occurring in between the camera-calibration stage and the maneuver stage, (or during the maneuver itself), the entire procedure may have to be repeated. Finally, while visual feedback may help to overcome some of these difficulties to some extent, if the grasped object is obscured by other objects during some part of the maneuver, it will be necessary to use an estimation of its image-plane location based on the calibration in order to derive the feedback signal. Further, as indicated above, in a task requiring the use of all 6 degrees of freedom, visual feedback may prove to be difficult to implement.

An alternative approach to vision-guided positioning is based on the technique of continually estimating the local nonlinear relationship from robot joint space to the image-plane location of certain points on the end-effector or on the grasped object during the maneuver itself, and is adopted in the method of "camera-space manipulation" suggested by Skaar[12] *et al.* This method treats a 3-dimensional rigid-body positioning problem in the following manner. The desired physical position and orientation of the grasped object are achieved when the grasped object occupies a specified position and orientation in the reference frame of the work-piece. The task comprises driving the robot joints based on the aforementioned transformation and a knowledge of the desired image plane locations of the points/features on the grasped object. These desired image-plane locations are computed on the basis of another estimated transformation—from the work-piece reference frame to the image-plane. Both transformations are estimated independently for each camera, and the maneuver objectives (i.e. the desired image-plane locations) are specified independently in the image plane of each camera. The relationship between the robot joint coordinates and the image-plane (camera-space) coordinates of a set of feature points on the grasped object is estimated by simultaneously sampling joint coordinates and camera-space coordinates of the observed points. Because the parameters in the estimation model are updated continually as the end-effector proceeds through the maneuver, the above relationship has the merit of being adaptive to external disturbances or intentional changes in camera location or orientation or internal parameters. Further, good local accuracy is possible with a simple camera model used together with the nominal kinematic model, in part because the procedure allows placing greater weights on samples taken closer to point of maneuver culmination. As a further benefit, the approach allows one to quantify and correct for imperfections in the end-effector grasp [13]. Also, the available redundancy in terms of the number of specified desired ("target") coordinates, number of cameras, as well as the factoring in of measurements of joint coordinates and camera-space coordinates made prior to maneuver culmination, allows for a terminal positioning precision that exceeds both the kinematic model precision and image-plane resolution.

The method of camera-space manipulation employs two or more independently mounted cameras in preference to an end-effector mounted camera (the "eye-in-hand" configuration). The use of that configuration requires an accurate kinematic model, a precise knowledge of the coordinate transformation between the camera mounted cartesian frame and the end-effector, and in general, also a mapping from image plane to the camera mounted cartesian reference frame. Possible changes in the camera field of view may necessitate frequent recalibration as the end-effector moves through the maneuver. Further, estimation of trajectories of moving targets (based on an idea of their possible cartesian space trajectories) is better accomplished with independent cameras

[12]. Finally, the work of Hackwood & Beni[14] has indicated that greater positioning precision is generally possible with 2 independently mounted cameras than with the eye-in-hand configuration.

The convenience of an adaptive estimation of the transformation (in the following two references, from physical space to image-plane coordinates) of points attached to the end-effector during the maneuver has also been utilized by Liang[15] *et al* and Chang & Liang[16]. These authors, however, specify and update the maneuver objectives in the robot mounted cartesian frame using the latest estimation of the transformation. The measurements supplied to the estimation algorithm are the 3-dimensional cartesian coordinates of points on the end-effector (in both the robot mounted reference frame and the camera mounted reference frames) and their corresponding image-plane coordinates. The precision available with this method is thus directly limited by the precision with which the robot kinematics are known. Moreover, the method requires stereo cameras and laser light to measure the latter of the above 3-D coordinates. Camera-space manipulation, on the other hand, allows robustness with respect to some types of kinematic model errors [13], does not need stereo cameras, and employs only ordinary spot-lighting.

2.GENERAL APPROACH AND TASK DESCRIPTION

The details of the method of camera-space manipulation for a 6 degree-of-freedom rigid-body positioning task are presented in [17]. The following is a brief description of the steps involved. The local, nonlinear relationship between the camera-space coordinates of points on the grasped object and the robot joint coordinate space is estimated independently for each camera, during the maneuver, based on the nominal kinematics of the robot and the orthographic camera model. Camera-space coordinates of certain easily extractable feature points ("cues") of the object grasped by the end-effector are used in conjunction with joint coordinate measurements for this purpose. The objective is to guide the cues on the grasped object ("manipulable object") in a least squares sense into their terminal target positions in two or more image planes using the latest estimates of parameters that characterize the above mapping from camera-space coordinates to joint-space. The knowledge of the desired position and orientation of the grasped object in the reference frame of the "non-manipulable object" (the work-piece, relative to which the former object is to be positioned) is utilized in computing the above target positions. These can be derived independently for each camera via an estimated transformation from the reference frame of the non-manipulable object to the image plane of that camera. When the camera-space coordinates of the cues on the manipulable object acquire the specified relationship relative to the cues on the non-manipulable object in two or more image planes, the manipulated object itself has acquired its desired position and orientation relative to the non-manipulable object in cartesian space [13].

The experimental work reported upon herein tests the method using a part mating task. Figure 1 shows the two members which are to be joined together. The tolerance of the fit may be varied in that the outer sleeve of the round insertion piece may be removed and replaced by a piece with a different outside diameter. The radius of curvature of each of the two approaching edges is essentially zero, so the role of passive compliance is very small. Because the rotation along the entering axis is specified and controlled, the task requires 6 degrees of freedom. A very large, 6-axis manipulator is used to perform the task.

The receiving piece consists of 76 $(\mathrm{mm})^3$ aluminum cube with a 25 mm hole drilled into the center of one side as shown in Figure 1. To enable speedy analysis of visual samples, readily detectable ring shapes, either dark on a light surface or light on a dark surface, are painted in precise locations onto this cube ("non-manipulable cues"). The experiment is set up such that, between the two participating cameras, at least four of the cube's surfaces are in view. The inside diameter of each ring is 6.5 mm, and the outside diameter is 19.5 mm. A large number of non-manipulable cues was chosen to take advantage of the positioning resolution that stems from this redundancy. The mix of black and white rings was chosen as a possible assist in the cue identification procedure [17].

The member that is inserted ("tool") consists of a small 38 $(\mathrm{mm})^3$ aluminum cube with a 38 mm-long cylinder of variable radius protruding from one side. "Manipulable cues" of the same size as the others are painted on the center of four of the small cube's surfaces as well as at the end of the cylinder. Only one of the rings on this manipulable body is black, while the rest are white [17].

The manipulator used is the 3,500 lb., six-axis GMF S-400, with AC servomotor actuation. Two standard black-and-white video cameras — Panasonic model WV-1410 — are connected to a Data Translation model DT-2851 frame-grabber board and Data Translation DT-2859 multiplexer board which are installed in a Zenith Z-248 computer. The Z-248 performs all image analysis, estimation and trajectory calculations. It supplies, at a rate of about one every 3 seconds, points along a joint-coordinate sequence to the controller of the GMF S-400, which in turn executes these commands with Cartesian interpolation, PID control, and AC Servomotor actuation.

3. EXPERIMENTAL RESULTS

Most of our experiments were performed using a sleeve size for the cylindrical tool that was 0.7 mm smaller (in radius) than that of the cylindrical hole of the receiving piece.

The most straightforward way of characterizing the results is in terms of the frequency of maneuver success. It was clear throughout our several hundred trials that the contribution of passive compliance to maneuver success was negligibly small. This claim is based first on the very small ($\leq$0.1 mm) radius of curvature of approaching edges, and second on the observed absence of movement of the receiving piece or relative shifting within the grasper of the entering piece during a successful insertion procedure. Although difficult to quantify, it was observed that the method typically produced a very accurate relative orientation of the two pieces before and during insertion. The effect of several of the operating factors on the frequency of maneuver success is summarized in the following (see [17] for details).

- While the precise locations of the cameras were never critical , adequate separation of the cameras was important. Focal axis lines separated by 85^0 to 95^0 worked well consistently. Using a 75 mm focal length lens on a standard black-and-white video camera, a distance of between 20 and 25 feet worked well.

- The parameters which are used to identify the local relationship between joint positions and the camera-space locations of manipulable cues were, in our tests, determined at two junctures during the maneuver. The first juncture was the end of a pre-planned trajectory. With these early estimates in hand — one set for each camera — the transition trajectory to the beginning of the straight-line final approach was determined [17]. At the end of the transition, estimates were updated based on measurements known to be close — both in terms of orientation and position — to the receiving cube. While these latter transition positions were weighted more heavily than those of the pre-planned trajectory, it was found consistently that the character of the pre-planned trajectory, and its terminal proximity to the cube, were critical to maneuver success. We observed that the motion should involve large ($\geq 20^0$) rotation of the wrist joints, that it should not include straight-line motion, and that a fairly large portion of it (perhaps the final 10 – 15 samples) should occur within reasonable translational proximity ($\sim$ 1 ft.) and rotational proximity ($\sim 5^0$ – 10^0) of the actual entering position.

- As for weighting of measurements, in these tests, it was found that a nearly equal weighting of all measurements, with only a slight skewing in favor of latter samples, worked best.

- The motivation for adding an additional camera was to assess the influence of this redundant third camera on error reduction in cases where the information from 2 cameras alone was inadequate. The focal axis of the third camera was separated from each of those of the other two by at least 90^0. Where the 2-camera error was 3 – 6 mm (the largest error we observed in any of our tests), the third camera nearly always reduced this error by at least a factor of two.

- We investigated partially the effect of the orthographic camera model by introducing to the raw camera-space measurements of the manipulable cues a distortion correction based on a pinhole camera model. The effect of this correction was always small but always beneficial if the error prior to applying it was more than two or three millimeters.

4. ESTIMATE-UNCERTAINTY PROPAGATION : NUMBER OF CAMERAS AND THEIR RELATIVE ORIENTATION

This section presents analytical work aimed at understanding some of the experimental findings described above. While several operating factors have been considered in the previous section, the objective here is to study how relative camera orientation and the number of participating cameras govern positioning precision, given the assumption that the parameter estimates (referred to as the C parameters, see [17]) available at the pre-terminal location contain normally distributed errors. It is assumed further that all cameras are equal in all respects and that the corresponding parameters of each camera contain the same amount of error. The point of interest here is to determine how the cameras may be oriented relative to each other and how the use of more than two cameras may help to restrict the propagation of these errors into the terminal physical space position of a point on the end-effector.

4.1 Analysis

In this work, a simplified camera-space manipulation task requiring three dimensional positioning of the tip of the end-effector of the 3-axis manipulator shown in Figure 2 is studied. The task represents substantially the positioning of a single manipulable cue. The estimation equations are here expressed in terms of cartesian coordinates rather than joint coordinates (see [12], [13], [14]), and to simplify calculations, the errors in estimates of C_j^l $(j = 1, \ldots, 6)$ are assumed to be uncorrelated in addition to being normally distributed about the correct values with small variances that are assumed *a priori*. The uncertainty of the parameter estimates is thus characterized by the assumed variances. If X_l^T and Y_l^T denote the target camera space location of the end-effector tip in the l^{th} camera space, and $f_x(x, y, z, \mathbf{C}^l)$ and $f_y(x, y, z, \mathbf{C}^l)$ (defined in Eqs. (11a) & (11b)) represent the local mapping from the robot mounted XYZ reference frame to the l^{th} camera space, and $\mathbf{C}^l$ denotes the latest estimates of the parameters C_j^l, then with L cameras, the desired terminal position of the end-effector tip is given by (See Junkins[18]),

$$\min_{x,y,z} J = \sum_{l=1}^{L} \left[X_l^T - f_x(x, y, z, \mathbf{C}^l)\right]^2 + \left[Y_l^T - f_y(x, y, z, \mathbf{C}^l)\right]^2 \tag{1}$$

The objective is to find the uncertainty in the terminal x, y, z evaluated from Eq. (1) above, given an uncertainty in each of the parameter estimates C_j^l. The terminal $\mathbf{r} = [x, y, z]^T$ is evaluated by solving the necessary conditions corresponding to the above minimization,

$$\frac{\partial J}{\partial x} = 0, \quad \frac{\partial J}{\partial y} = 0, \quad \frac{\partial J}{\partial z} = 0, \tag{2}$$

These yield the following nonlinear relations,

$$g_1(x, y, z, C_1^1, C_2^1, \ldots, C_6^L) = 0 \tag{3}$$

$$g_2(x, y, z, C_1^1, C_2^1, \ldots, C_6^L) = 0 \tag{4}$$

$$g_3(x, y, z, C_1^1, C_2^1, \ldots, C_6^L) = 0 \tag{5}$$

Each camera is associated with 6 parameters, so that the functions $g_i (i = 1 \; to \; 3)$ represent hypersurfaces in $6L + 3$ dimensions. Therefore,

$$\Delta g_i = \frac{\partial g_i}{\partial x}\Delta x + \frac{\partial g_i}{\partial y}\Delta y + \frac{\partial g_i}{\partial z}\Delta z + \frac{\partial g_i}{\partial C_1^l}\Delta C_1^l + \cdots + \frac{\partial g_i}{\partial C_6^L}\Delta C_6^L \; = \; 0 \; for \; i = 1 \; to \; 3 \quad (6)$$

Written in matrix form this yields,

$$\mathbf{A}\Delta\mathbf{r} \; = \; -\mathbf{B}\Delta\mathbf{C} \quad (7)$$

where

$$\mathbf{A} = \begin{bmatrix} \frac{\partial g_1}{\partial x} & \ldots & \frac{\partial g_1}{\partial z} \\ & \ldots & \\ \frac{\partial g_3}{\partial x} & \ldots & \frac{\partial g_3}{\partial z} \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} \frac{\partial g_1}{\partial C_1^l} & \ldots & \frac{\partial g_1}{\partial C_6^L} \\ & \ldots & \\ \frac{\partial g_3}{\partial C_1^l} & \ldots & \frac{\partial g_3}{\partial C_6^L} \end{bmatrix}$$

and,

$$\Delta\mathbf{r} = [\Delta x, \Delta y, \Delta z]^T \; and \; \Delta\mathbf{C} = [\Delta C_1^1, \ldots, \Delta C_6^L]^T$$

Whenever**A** is nonsingular,

$$\Delta\mathbf{r} \; = \; -\mathbf{A}^{-1}\mathbf{B}\Delta\mathbf{C} \quad (8)$$

Multiplying both sides by $\Delta\mathbf{r}^T$,

$$\Delta\mathbf{r}\Delta\mathbf{r}^T \; = \; \mathbf{A}^{-1}\mathbf{B}\Delta\mathbf{C}\Delta\mathbf{C}^T\mathbf{B}^T\mathbf{A}^{-T} \quad (9)$$

Now assuming that both sides of Eq. (9) are evaluated for a number of $\Delta\mathbf{C}$, one can compute an expected $\Delta\mathbf{r}\Delta\mathbf{r}^T$ such that,

$$E(\Delta\mathbf{r}\Delta\mathbf{r}^T) \; = \; \mathbf{A}^{-1}\mathbf{B}E(\Delta\mathbf{C}\Delta\mathbf{C}^T)\mathbf{B}^T\mathbf{A}^{-T} \quad (10)$$

or,

$$\mathbf{\Lambda}_{rr} \; = \; \mathbf{A}^{-1}\mathbf{B}\mathbf{\Lambda}_{CC}\mathbf{B}^T\mathbf{A}^{-T}$$

where $\mathbf{\Lambda}_{rr}$ and $\mathbf{\Lambda}_{CC}$ denote the covariance matrices corresponding to **r** and **C**, where $\mathbf{C} \equiv [C_1^1, \ldots, C_6^L]^T$. Eq. (10) relates the uncertainty of the estimates of parameters C_j^l as it propagates to the terminal position **r**. The functions f_x and f_y for each camera are of the form,

$$X_l = \left(C_1^{l^2} + C_2^{l^2} - C_3^{l^2} - C_4^{l^2}\right)x + 2\left(C_2^l C_3^l + C_1^l C_4^l\right)y + 2\left(C_2^l C_4^l - C_1^l C_3^l\right)z + C_5^l \equiv f_x(x, y, z, \mathbf{C}^l) \quad (11a)$$

$$Y_l = \left(C_2^l C_3^l - C_1^l C_4^l\right)x + 2\left(C_1^{l^2} - C_2^{l^2} + C_3^{l^2} - C_4^{l^2}\right)y + 2\left(C_3^l C_4^l - C_1^l C_2^l\right)z + C_6^l \equiv f_y(x, y, z, \mathbf{C}^l) \quad (11b)$$

The parameters $C_1^l, \ldots, C_m^l$, $m \; = \; 4$, are related to the Euler parameters corresponding to the camera l's orientation relative to the robot reference frame XYZ, by,

$$e_0 = C_1^l / \|C^l\|, \; e_1 = C_2^l / \|C^l\|, \; e_2 = C_2^l / \|C^l\|, \; e_3 = C_3^l / \|C^l\| \quad (12)$$

where $\|C^l\| \; = \; C_1^{l^2} + C_2^{l^2} + C_3^{l^2} + C_4^{l^2}$ depends on camera focal length, digitization scaling, etc, and the distance of separation between the camera and the end-effector tip, measured along the camera optical axis. The direction cosine matrix corresponding to a given camera orientation is given by,

$$[c^l] = \begin{bmatrix} e_0{}^2 + e_1{}^2 - e_2{}^2 - e_3{}^2 & 2(e_1e_2 + e_0e_3) & 2(e_1e_3 - e_0e_2) \\ 2(e_1e_2 - e_0e_3) & e_0{}^2 - e_1{}^2 + e_2{}^2 - e_3{}^2 & 2(e_2e_3 + e_0e_1) \\ 2(e_1e_3 + e_0e_2) & 2(e_2e_3 - e_0e_1) & e_0{}^2 - e_1{}^2 - e_2{}^2 + e_3{}^2 \end{bmatrix} \qquad (13)$$

where, in terms of the 3-1-3 Euler angles,ϕ, θ and ψ,

$$e_0 = \cos\left(\frac{\phi+\psi}{2}\right)\cos\left(\frac{\theta}{2}\right) \qquad (14a)$$

$$e_1 = \cos\left(\frac{\phi-\psi}{2}\right)\sin\left(\frac{\theta}{2}\right) \qquad (14b)$$

$$e_2 = \sin\left(\frac{\phi-\psi}{2}\right)\sin\left(\frac{\theta}{2}\right) \qquad (14c)$$

$$e_3 = \sin\left(\frac{\phi+\psi}{2}\right)\cos\left(\frac{\theta}{2}\right) \qquad (14d)$$

4.2. Results

In these calculations, $\|C'^l\|$ was assumed to be equal for all participating cameras, and a value of $\|C^l\| = 0.24$ was used. C_5^l and C_6^l were chosen in each case to make the image-plane location of the end-effector tip coincide with the image-plane center. In the study of the relative orientation of two cameras, the Euler angles of camera 1 were assumed to be $\phi = 90^0$, $\theta = 34^0$, and $\psi = -90^0$ (In the context of Figure 2, $\alpha = 0^0$ and $\beta = 0^0$). The orientation of camera 2 was obtained by premultiplying the $[c^1]$ by,

$$l_r = \begin{bmatrix} \cos(\alpha) & 0 & \sin(\alpha) \\ 0 & 1 & 0 \\ -\sin(\alpha) & 0 & \cos(\alpha) \end{bmatrix} \qquad (15)$$

where α was allowed to vary from 0^0 to 180^0 . The covariance $\mathbf{\Lambda}_{CC}$ was assumed to be $\mathbf{\Lambda}_{CC} = diag[0.000045, \ldots, 0.000045]$.

Figure 3 displays the sum of the diagonal elements of $\mathbf{\Lambda}_{rr}$, $E(\Delta x^2 + \Delta y^2 + \Delta z^2)$ as a function of α, and shows the propagated variance to be minimum near $\alpha = 90^0$, but not significantly greater over the range of about 60^0 about this angle, which effect was also observed in the rigid-body positioning experiments. Figures 4 and 5 compare the region of physical space uncertainty caused by the uncertainty in each camera's C_1 through C_4 for the 2 cases when $\alpha = 5^0$ and $\alpha = 90^0$. The case with $\alpha = 0^0$ yields a condition that the position ambiguity along the camera optical axis of neither camera is eliminated by the presence of the other, giving rise to infinite estimation uncertainty.

The effect of the addition of a third camera was studied as follows. Assuming that the two original cameras were oriented at $\alpha = 90^0$, a 3^{rd} camera was introduced (with the same variance in its C_j elements as the other two) at orientations ranging from $\alpha = 0^0$ to 90^0 and $\beta = 0^0$, and next at orientations ranging from $\beta = 0^0$ to 90^0, keeping $\alpha = 60^0$. In all cases, the addition of the 3^{rd} camera improved the positioning precision as observed from the value $E(\Delta x^2 + \Delta y^2 + \Delta z^2)$. However, it was also seen that the 3^{rd} camera could be added at a certain orientation to minimize the uncertainty in a specific direction if this was required in a given maneuver. This effect is shown in Figures 6 and 7. The results support the experimental finding that the addition of the 3^{rd} camera improved positioning precision when it was placed at an angle of 90^0 relative to the optical axes of the original two cameras.

While the above results do provide support to the relevant experimental observations, it is important to note the following. (1) The analysis assumes equal covariances on $\mathbf{C}^l$ independent of the camera orientation. In practice, the $\mathbf{C}^l$ estimate accuracy and camera perspective will not be independent since the number of manipulable cue samples used to determine $\mathbf{C}^l$ will vary with camera perspective. (2) The underlying source of error in the $\mathbf{C}^l$ estimates is twofold : First, there is "random" error in the camera-space measurements of those manipulable cues which are sampled during the approach. Second, there is some non-negligible ("deterministic") error in both the camera model and the kinematic model which serve as the basis for the estimates. There is no distinction between the two sources of error in the analysis, but instead, both are included in an overall assumed uncertainty distribution. (3) Furthermore, the analysis assumes that the expected values of all $6L$ parameters are compatible with exact resolution of those joint rotations which culminate the maneuver correctly. Implicit in this assumption is that error in the target camera-space positions is negligible, and that, locally, error in the kinematic model is also negligible.

5. SUMMARY AND CONCLUSION

The ability of camera-space manipulation to produce very accurate, tight-tolerance, vision-based assembly without carefully-controlled, calibrated camera positioning is shown. The camera and receptacle positions were arbitrary, but within bounds, in these experiments.

An uncertainty analysis of a three-dimensional point placement task showed that a relative camera orientation of 90^0 minimized propagation of random errors from estimated parameters to the commanded terminal position, and that comparable precision could still be obtained for a range of relative orientation from 60^0 to 120^0. The use of a third camera was shown to be beneficial both in reducing the uncertainty in the absolute terminal position, as well as in controlling the uncertainty in specific directions. Both observations are consistent with the findings of the experimental study.

ACKNOWLEDGMENTS

This work has been supported by the U.S. Office of Naval Research under grant no. N00014-91-J-1054. The authors are grateful to Dr. R.K. Miller of Iowa State University for his contribution to the software development.

REFERENCES

[1] Tsai, R.Y., 'Synopsis of Recent Progress on Camera Calibration for 3D Machine Vision', *The Robotics Review*, Eds : Khatib, O. *et al*, MIT Press, pp. 146–159, 1989.

[2] Ishii, M., Sakane, S., Kakikura, M., and Mikami, Y., 'A 3-D Sensor System for Teaching Robots Paths and Environments', Int. J. Robotics Research, Vol. 6, No. 2, pp. 45–59, 1987.

[3] Abidi, M.A. and Gonzalez, R.C.,'The Use of Multisensor Data for Robotic Applications', *IEEE Trans. Robotics & Automation*, Vol. 6, No. 2, pp. 159–177, 1990.

[4] Mochizuki, J., Takahashi, M., and Hata, S.,'Unpositioned Workpieces Handling Robot with Visual and Force Sensors', *IEEE Trans. Industrial Electronics*, Vol. IE-34, No. 1, pp. 1–4, 1987.

[5] Agapakis, J.E.,'Approaches for Recognition and Interpretation of Workpiece Surface Features Using Structured Lighting', *Int. J. Robotics Research*, Vol. 9, No. 5, pp. 3–16, 1990.

[6] Shirai, Y. and Inoue, H., 'Guiding a Robot by Visual Feedback in Assembling Tasks', *Pattern Recognition*, Vol. 5, pp. 99–108, 1973.

[7] Tani,K., Abe, M., Tanie, K. and Ohno, T.,'High Precision Manipulator with Visual Sense', *Proc. 7 th Int. Symp. on Industrial Robots : Japan Industrial Robot Association*, Tokyo, pp. 561–568, 1977.

[8] Weiss, L.E., Sandersen, A.C. and Neuman, C.P.,'Dynamic Sensor-Based Control of Robots with Visual Feedback', *IEEE J. of Robotics & Automation*, Vol. RA-3, No. 5, pp. 404–417, 1987.

[9] Feddema, J.T. and George Lee, C.S.,'Adaptive Image Feature Prediction & Control

for Visual Tracking with a Hand–Eye Coordinated Camera', *IEEE Trans. on Systems, Man, & Cybernetics*, Vol. 20, No. 5, pp. 1172–1183, 1990.

[10] Hollerbach,J.M.,'A Survey of Kinematic Calibration', *Robotics Review 1*, Ed: Khatib *et al*, MIT Press, pp. 207–242, 1989.

[11] Horn. B.K.P.T.,*Robot Vision*, McGraw-Hill Book Co., Ch. 13, 1986.

[12] Skaar, S.B., Brockman W.H. and Hanson, R.,'Camera Space Manipulation', *Int. J. Robotics Research*, Vol . 6, No. 4, pp. 20–32, 1987.

[13] Skaar, S.B., Brockman, W.H. and Jang, W.S.,'Three Dimensional Camera Space Manipulation', *Int. J. Robotics Research*, Vol. 9, No. 4, pp. 22–39, 1990.

[14] Hackwood, S., and Beni., G.,'Sensor and High Precision Robotics Research', *Robotics Research*, Eds : M. Brady & R. Paul, MIT Press, pp. 529–554, 1984.

[15] Liang, P., Chang, Y.L., and Hackwood, S.,'Adaptive Self-Calibration of Vision-Based Robot Systems', *IEEE Trans. Systems, Man & Cybernetics*, Vol. 19, No. 4, pp. 811–823, 1989.

[16] Chang, Y.L. and Liang, P.,'On Recursive Calibration of Cameras for Robot Hand–Eye Systems', *Proc. IEEE Int. Conf. on Robotics & Automation*, pp. 838–843, 1989.

[17] Skaar, S.B., Chen, W.Z. and Miller, R.K.,'High Resolution Camera Space Manipulation', *Proc. ASME Design Automation Conf.*, Miami, Sept., 1991.

[18] Junkins, J.L.,*An introduction to Optimal Estimation of Dynamical Systems*, Alphen Aaan Den Rijn, Sitjthoff & Noordhoff, pp. 29–33, 1978.

for Visual Tracking with a Hand-Eye Coordinated Camera', *IEEE Trans. on Systems, Man, & Cybernetics*, Vol. 20, No. 5, pp. 1172–1183, 1990.
[10] Hollerbach,J.M.,'A Survey of Kinematic Calibration', *Robotics Review 1*, Ed: Khatib *et al*, MIT Press, pp. 207–242, 1989.
[11] Horn. B.K.P.T.,*Robot Vision*, McGraw-Hill Book Co., Ch. 13, 1986.
[12] Skaar, S.B., Brockman W.H. and Hanson, R.,'Camera Space Manipulation', *Int. J. Robotics Research*, Vol . 6, No. 4, pp. 20–32, 1987.
[13] Skaar, S.B., Brockman, W.H. and Jang, W.S.,'Three Dimensional Camera Space Manipulation', *Int. J. Robotics Research*, Vol. 9, No. 4, pp. 22–39, 1990.
[14] Hackwood, S., and Beni., G.,'Sensor and High Precision Robotics Research', *Robotics Research*, Eds : M. Brady & R. Paul, MIT Press, pp. 529–554, 1984.
[15] Liang, P., Chang, Y.L., and Hackwood, S.,'Adaptive Self-Calibration of Vision-Based Robot Systems', *IEEE Trans. Systems, Man & Cybernetics*, Vol. 19, No. 4, pp. 811–823, 1989.
[16] Chang, Y.L. and Liang, P.,'On Recursive Calibration of Cameras for Robot Hand-Eye Systems', *Proc. IEEE Int. Conf. on Robotics & Automation*, pp. 838–843, 1989.
[17] Skaar, S.B., Chen, W.Z. and Miller, R.K.,'High Resolution Camera Space Manipulation', *Proc. ASME Design Automation Conf.*, Miami, Sept., 1991.
[18] Junkins, J.L.,*An introduction to Optimal Estimation of Dynamical Systems*, Alphen Aaan Den Rijn. Sitjthoff & Noordhoff, pp. 29–33, 1978.

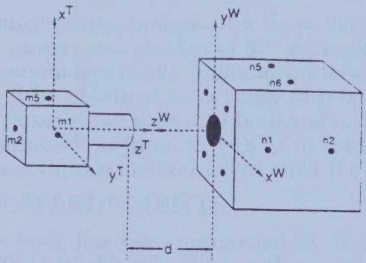


Figure 1: Cue Positions on the Entering and Receiving Pieces.

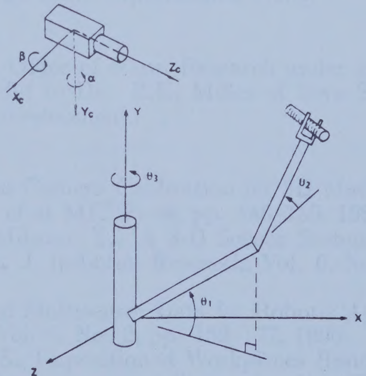


Figure 2: Three Dimensional Point Placement.

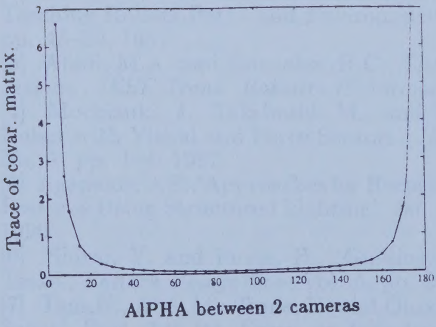


Figure 3: Uncertainty Variation with Relative Orientation α Between 2 Cameras.

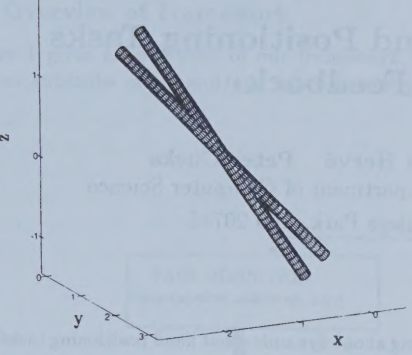


Figure 4: Region of Uncertainty with 2 Cameras at $\alpha = 10^0$.

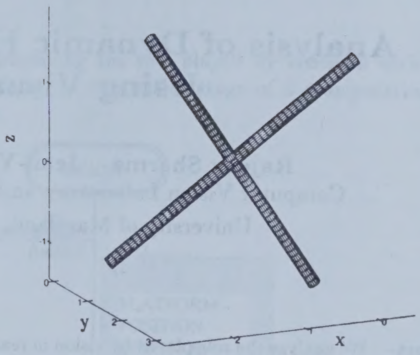


Figure 5: Region of Uncertainty with 2 Cameras at $\alpha = 90^0$.

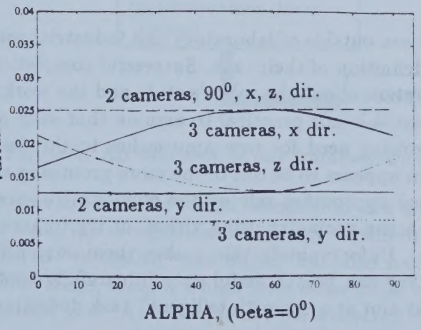


Figure 6: Uncertainty Variation for 3 Cameras (α : Variable, $\beta = 0^0$)

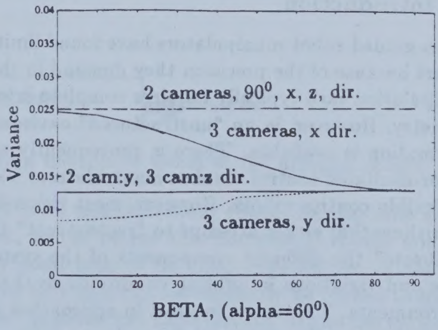


Figure 7: Uncertainty Variation for 3 Cameras (β: Variable, $\alpha = 60^0$)

Analysis of Dynamic Hand Positioning Tasks Using Visual Feedback

Rajeev Sharma **Jean-Yves Hervé** **Peter Cucka**
Computer Vision Laboratory and Department of Computer Science
University of Maryland, College Park, MD 20742

Abstract— We analyze the role played by vision in reasoning about dynamic robot hand positioning tasks, such as catching, hitting, and interception, that also involve an object (target) moving in the vicinity of the robot. We present a framework for qualitative reasoning about the corresponding hand/eye coordination problem and demonstrate how robust vision-based control may also be realized by tracking of image features using a Kalman filter. We examine the different levels at which visual input is involved in pursuing the dynamically-defined goal. A given task is transformed into one of constrained trajectory planning on a topological space defined by a set of observable image parameters. By considering only the qualitative features of the *perceptual control surface* and by properly defining the tasks in terms of this surface we remove the need for precise calibration of different parts of the system. This differs significantly from the classical approaches that require more accurate description of the robot environment and the manipulation task.

1. Introduction

Vision-guided robot manipulators have found limited use outside of laboratory and industrial settings, in part because of the precision they demand in the definition of their task. Successful completion of a manipulation task typically requires complete information about the robot's state and the work space geometry. However, in an "unstructured" environment it is not practical to assume that such precise information is available. There is consequently a growing need for new approaches to the design of sensor-mediated control of robot manipulators. Vision appears to be one of the more promising sensors for flexible control robots. However, most vision-based approaches rely on the quantitative success of algorithms that either attempt to "reconstruct" the scene geometry using vision, or try to accurately "calibrate" the different components of the system. Unfortunately this makes them susceptible to noise and variations in lighting conditions so that they can be successful only in carefully-controlled environments. This raises interest in approaches that aim at a more "intelligent" task definition such that a more qualitative and informed use of visual input can be made.

We examine dynamic hand-positioning tasks involving visual feedback, and manipulation of a moving target. Such tasks include, for example, catching, hitting, and intercepting a moving object. We present a framework that transforms the task into one of trajectory planning and control in terms of a topological surface, called the *perceptual control surface*, using a camera space mapping introduced in [8] (see also [18]). Recognizing that vision is the main bottleneck in such real-time systems, our goal is robust and efficient use of visual data, and is achieved by exploiting non-local topological properties

---

The financial support of the Defense Advanced Research Projects Agency (ARPA Order No. 6989), U.S. Army Engineer Topographic Laboratories (Contract DACA76-89-C-0019), and IBM (Graduate Fellowship for the first author) is gratefully acknowledged.

of the perceptual control surface instead of an exact mapping that would be difficult to realize in practice. As an added benefit, the need for the cumbersome calibration techniques used in classical approaches is decreased.

2. Overview of Framework

Figure 1 gives an overview of our framework, emphasizing the role played by vision in each of the various subtasks of a hand/eye coordination problem. In the initial stage of a manipulation task,

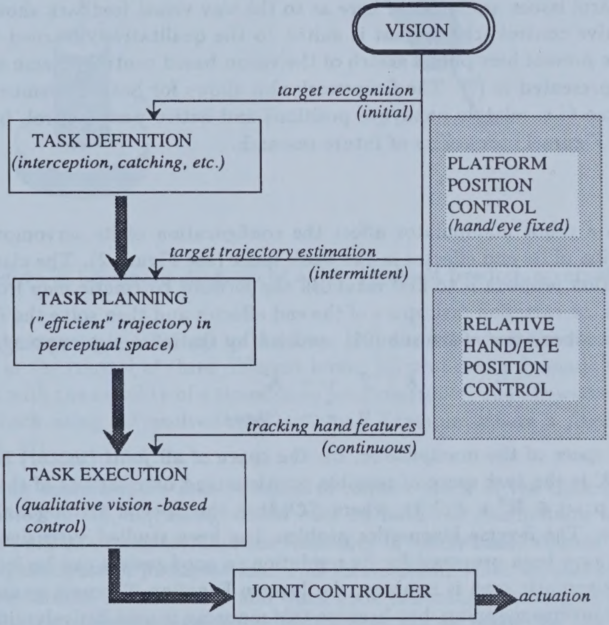

VISION
target recognition
(initial)
TASK DEFINITION
(interception, catching, etc.)
PLATFORM POSITION CONTROL
(hand/eye fixed)
target trajectory estimation
(intermittent)
TASK PLANNING
("efficient" trajectory in perceptual space)
RELATIVE HAND/EYE POSITION CONTROL
tracking hand features
(continuous)
TASK EXECUTION
(qualitative vision -based control)
JOINT CONTROLLER
actuation

Figure 1: An overview of our framework showing the role of vision at the different stages.

visual cues are used for target recognition and a high level task definition. Given the real-time nature of the task, the recognition is strongly model-driven, to the extent that the system should not only be able to "recognize" the object but also have a good idea of its size. The output of the *task definition* stage, then, is a high-level description of the task that can subsequently be refined to specify a kinematic goal. An example of such a description is "position the hand directly across the trajectory of the moving target while maintaining the minimum possible velocity at the time of contact." The task is defined in terms of the parameters of the size of the object and a qualitative notion of the orientation of the hand relative to the object at the time of contact.

At the next level, vision is used for estimating the target trajectory. This, in turn, is used in generating a trajectory plan for the robot that would achieve the goal prescribed in the task definition stage while ensuring that no dynamic constraints of the robot are violated and that the relevant costs (time, distance, energy expended, *etc.*) are optimized. To transform the general task definition into a plan we need a translation schema, which at this stage we envisage as comprising a set of rules that can capture the "intelligence" involved in classes of manipulation tasks, as we discuss later. Since we deal with more qualitatively defined tasks and representations of the robot kinematics that are imprecise (though stable in the relevant control space)—replanning or updating of plans would be necessary. However, the issue of the number of updates of the plan needs to be addressed with respect to time constraints, because repeated updating of plans would be slow and would make the execution and control more difficult. A two-stage or single-correction *task planner* is therefore envisioned.

The last stage involves the generation of the sequence of controls to the robot for following the planned trajectory. Here, vision plays the role of providing continuous feedback for tracking the hand position. Several control issues are involved here as to the way visual feedback should be used ([15]). We present an adaptive control scheme that is suited to the qualitatively-learned kinematics of the hand/eye system. We present here only a sketch of the vision-based control scheme used. More details of our approach are presented in [7]. The framework also allows for both movement of the hand/eye system with fixed pose (*i.e.* relative hand/eye position) and active pose control, both of which (see the shaded blocks of Figure 1) are topics of future research.

3. Background

The controls sent to a robot manipulator affect the configuration of its servomotors and thus the position and orientation of its end effector in the work space (see Figure 2). The classical approach to solving the manipulation problem is to first establish the *forward kinematic map* from the joint space of the manipulator to the cartesian task space of the end effector and then solve the *inverse kinematics problem.* The forward kinematics of the robot is modeled by the following mapping:

$$\begin{array}{rcl} \kappa : \mathcal{J} & \longrightarrow & \mathcal{K} \\ \mathbf{q} & \longmapsto & (\mathbf{p}, \omega) \end{array}$$

where $\mathcal{J}$ is the *joint space* of the manipulator, *i.e.* the space of all *joint* (motor) *configurations* $\mathbf{q} = (q_1, q_2, \ldots, q_n)'$, and $\mathcal{K}$ is the *task space* of possible positions and orientations of the end effector, *i.e.* the set of all pairs $(\mathbf{p}, \omega) \in \mathbb{R}^3 \times \mathcal{SO}(3)$, where $\mathcal{SO}(3)$ is the Special Orthogonal Group of three-dimensional rotations. The inverse kinematics problem has been studied extensively, and numerous models and methods have been proposed for its resolution (a good review can be found in [9]). When the Jacobian of the kinematic map is nonzero, the Inverse Function Theorem guarantees locally the existence of a smooth inverse mapping, but because this mapping is prohibitively difficult to compute, inverse kinematics algorithms generally provide only a fast approximation to the solution. When the Jacobian is zero (*i.e.* $\mathbf{q}$ is a singular point of the kinematic map), the inversion is even more complex, if not impossible, but it generally suffices to determine the location of these singularities and to try to avoid them.

Incorporating visual feedback into classical robot control leads to the *visuo servo control* problem. A good introduction on the different mechanisms of visual feedback involved in visuo servo control can be found in [15]. In particular, an important distinction made is that of the feedback representation mode, which can be either *position-based* or *image-based.* The position-based approach basically involves the problem of "visual reconstruction" which is recognized to be difficult because of the inherent non-linearities in the transformation and the uncertainties in the imaging process. On the other hand, image-based approaches require more work in task specification in the image (camera) space. We take an image-based approach in this paper, though we deviate from the control mechanism characterized in [5, 12, 15, 20].

Another important distinction, regarding the design and use of visual feedback is whether the camera is moving (being mounted on the robot arm) or fixed. Both [20] and [5] consider the *eye-in-*

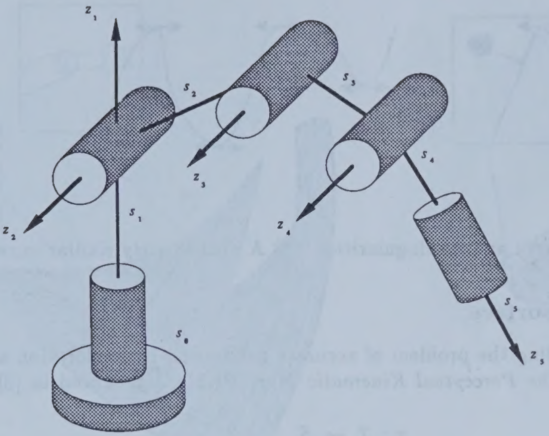


Figure 2: Schematic diagram of a five-degree-of-freedom manipulator

hand case, when the camera is attached to the arm. [20] also considers the hierarchical organization of visual feedback in the control at three different levels. [5] proposes a dynamic feature-based system similar to [20], but with the stability of a closed-loop joint controller, and allows for variable-time delays in the visual feedback using a "resolved-rate" schema. Examples where a *fixed-eye* configuration is used are [1, 12, 16, 17].

The idea of using camera-space manipulation to remove much of the difficulties involved in calibration was exploited in [16, 17], using visible cues on both the manipulator and the object being manipulated. The work also demonstrated the feasibility of vision-based control without prior knowledge of the camera/manipulator juxtaposition. This formulation, which we also adopt, would be useful particularly in unstructured environments and when good calibration techniques are not available (see [9, 19]). Assuming that easily-distinguishable visible cues can be placed on the object being manipulated ([16]) restricts one to mainly industrial applications, and would need to be relaxed under the more general framework of manipulation that we are considering.

To achieve more intelligent robot control, and to steer away from very precise techniques suitable only for industrial applications ([13]), learning must also be incorporated into the framework. In [16, 17] a more classical approach is used to learn the camera space kinematics—in which the parametric representation of the kinematic transformation is first simplifies. Considerable success has also been reported, in limited settings (*e.g.* grasping objects on a conveyor belt), by neural-network-based approaches ([10, 11]) that learn the kinematic map without explicit representation of the different transformations. [10] particularly , is in agreement with the underlying philosophy of behavior-based robotics (see [3]). We take the middle ground with respect to learning by choosing a "qualitative" representation of the kinematic map, rather than either representing it in an exact parametric form (with the parameters to be learned) or ignoring the underlying analytical structure and considering just the input-output relationships.

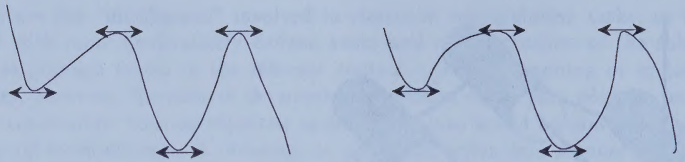


Figure 3: (a) A curve and its singularities. (b) A qualitatively similar curve.

4. Perceptual Control Surface

We propose a method of avoiding the problem of accurate calibration by exploration and learning of the qualitative properties of the *Perceptual Kinematic Map* (PKM), introduced in [8], which is the mapping

$$\pi : \mathcal{J} \rightarrow \mathcal{S}$$

where $\mathcal{S}$ is a set of measurable image parameters $\mathbf{s} = (s_1, s_2, \ldots, s_n)^T$ corresponding to features on the image of the manipulator. For example, if the robot rotates one of its joints while holding all others stationary, we expect that a given point on its hand will trace an arc of a circle in the workspace, and therefore an arc of an ellipse in the image plane. We can take this geometric interpretation a step further by considering the parametric hypersurface that is implicitly defined by the PKM. We call this surface the *Perceptual Control Surface* (PCS). The exact shape of the control surface depends on the camera position and orientation, which we do not know. However, the general analytic form of the PKM is available, allowing us to study the topology of the control surface and to know at least what it *can* look like. In particular, we are interested in the singularities of the PKM, not only because, as with techniques relying on calibration, we would like eventually to be able to avoid them, but also because, in the meantime, they tell us about the shape of the control surface. As a simple example, consider a function of one variable whose singular points (extrema) have been detected (Figure 3(a)). There are obviously an infinity of curves admitting the marked points as singularities, but all these curves will be *qualitatively* similar (Figure 3(b)). This principle can be extended to higher dimensions, allowing us to say that a surface is *qualitatively* uniquely defined by its singularities. Figure 4a illustrates a partial PCS and is a parametric plot of s_1, s_2 and q_1 as a function of q_1 and q_2. The two image parameters chosen are the X and Y coordinates of a corner point for a white rectangle marked on the hand. These parameters are adequate for our analysis of the general performance of our model, but in a more realistic setting other features, such as the center point of the image of the hand, may be more suitable. Thus two positions of the hand map to two different points (P_1 and P_2) on the PCS (Figure 4a).

A major requirement for our geometric approach to be feasible is good repeatability of joint motion. We do not need to know the absolute value of the joint angles, and in fact, we do not even need to have one-to-one correspondence between joint parameters and physical joints, which makes our representation valid even when—as is often the case—several joint parameters have to be controlled in order to obtain, pure motion of a single link. An important task that forms the basis of control is distinguishing between the *singular* and *regular* portions of the PCS. The singularities of the PCS need to be identified so that they may be avoided during manipulation. At the non-singular points of the PCS, the surface may be approximated by the local tangential plane, so that an adequate control strategy can be defined by a simple gradient descent on a cost function defined for the task. The

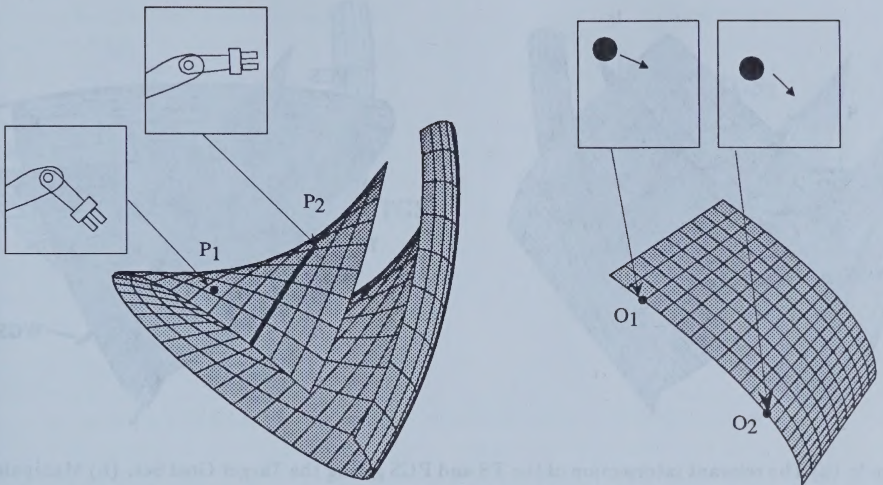


Figure 4: (a) Positions of the manipulator mapped onto the PCS. (b) Target Surface (TS) associated with a moving target.

qualitative properties of the PCS, such as the approximate location and type of a singularity, can be learned by repetitive exploration, as outlined in [7]. We present the details of the learning paradigm elsewhere. Here we assume that a qualitative representation of the PCS is available and can form a basis for manipulator control.

5. Redefining Hand-Positioning Tasks

With each moving target we identify the goal set in the image that corresponds to the hand in the desired state for manipulation (grasping, hitting, *etc.*). The profile of this goal set with time defines a *target surface* (TS) in the camera space and can be studied independent of the PCS of a particular hand/eye system. Figure 4b gives an example of the target surface for the simple task of grasping a ball moving along a parabolic trajectory. In general the surface will be more complex and will represent the variety of positions that accomplish the same task. The generation of a goal set corresponding to a grasping task for a given object is considered in detail in [4]. To simplify the discussion of the target surface, we assume that a single point/feature defines the grasping goal. If the target's motion is not random and is slowly changing, a target surface may be defined by, for example, fitting a second order interpolant to the visual data for the moving target. For a fixed robot base and fixed camera (*i.e.* a fixed PCS), intelligent manipulation requires an analysis of the goal while the target is in the workspace of the robot. This is achieved through a study of the intersection of the TS and the PCS. It requires rough knowledge of the location of the target surface, which can be obtained using a stereo camera or *a priori* knowledge of the target size. The PCS of Figure 4a and the TS of Figure 4b intersect in Figure 5a to define a *Target Goal Set* (TGS), with t_i and t_f defining the initial and final times for which the manipulation can take place. Thus the entire manipulation task must be carried out in the interval τ, $[t_i, t_f]$. In general, there may be several such intervals—but for simple target trajectories we can assume a single interval. The estimation of τ and the TS also takes into account the minimum time required for the the manipulator to decelerate before reaching its joint limits.

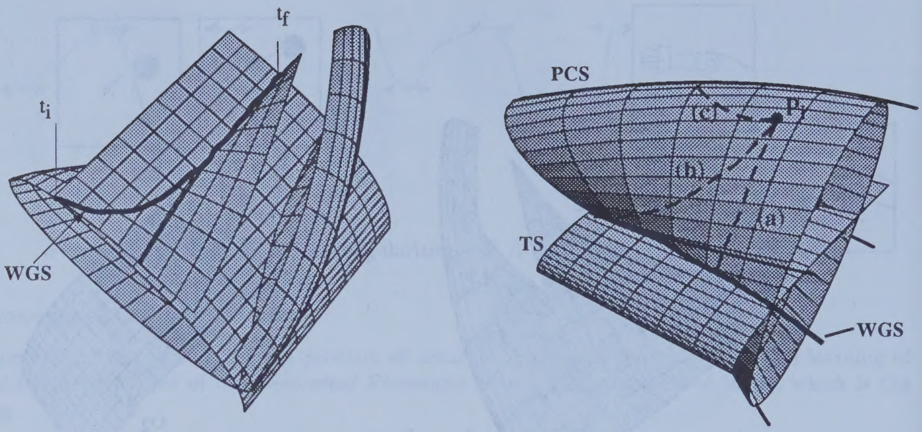


Figure 5: (a) The relevant intersection of the TS and PCS giving the Target Goal Set. (b) Manipulation tasks defined as trajectory planning on the PCS.

The dynamic manipulation problem then reduces to that of trajectory planning on the PCS from the initial position of the manipulator to some goal position g on the TGS. The planning would consist of first deciding whether such a g exists, *i.e.* whether the task is feasible, and, if so, then determining the g and a corresponding path on the PCS for accomplishing the task. Though the actual path would depend on the particular task we can identify the following generic tasks for a robot manipulator involving a moving object in its vicinity: (a) *Interception:* position the hand in the path of the moving object so that at some point the object hits the hand. (b) *Pursuit:* move the hand to pursue and grasp the moving object with the lowest possible relative velocity at the time of contact. (c) *Evasion:* move the hand to avoid being hit by the moving object. Each of these tasks may be defined as a variation of a trajectory planning problem on the PCS, and an example path for reach is given in Figure 5b.

The trajectory planning to achieve the above tasks is done by gradient descent on a cost function $\mathcal{D}$ that is a function of the current state of the robot (as tracked by a Kalman filter, see [7]) and the current target goal set (TGS) and is designed to reflect the characteristics of the path as defined by the current task. For example, if the goal is (b) above, then the cost function must include a term that gives the current estimated difference between the hand and object velocities, so that the difference may be reduced. The cost function must be minimized subject to the kinematic constraints of the robot, which in turn reflect its worst case dynamic performance (see [14]). The planner should thus have, for each joint i, the velocity and acceleration bounds, V_i and A_i respectively, such that the joint parameters satisfy the constraints $q_i' < V_i$, and $q_i'' < A_i$.

A critical design issue is the number of 'corrections' of the goal definition, or indeed whether the goal will be defined continuously in a dynamic manner ([15]). Although our framework allows for continuous updating of the target goal set, if the movement of the target is constrainted (as discussed before), a *single correction* strategy will suffice, simplifying the trajectory planning since the goal set would be updated only once. The single correction strategy would also lead to a better dynamic performance since the visual feedback involved in tracking the target will not introduce a bottleneck. However, the target would still be tracking—both for providing the single correction and for a warning

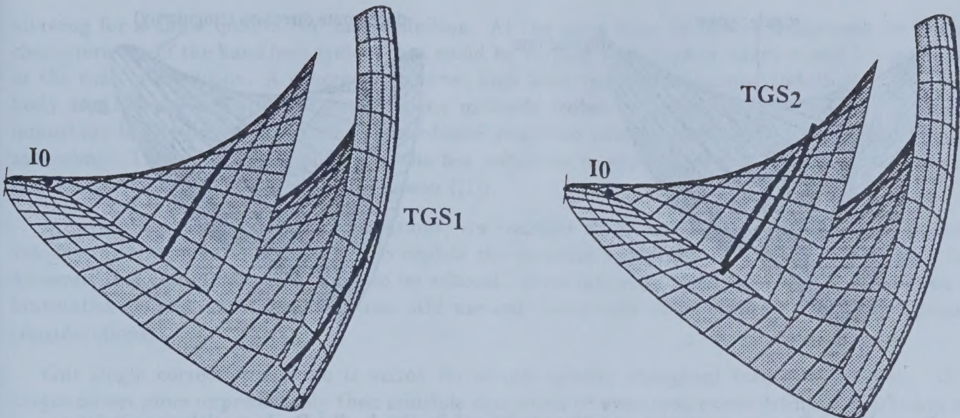


Figure 6: Movement of observer (hand-eye fixed) used to improve the target goal set.

if the target trajectory disagrees grossly with the particular dynamic model being used. The latter could be used for example for learning the behavior of the target in case the particular task is repeated.

With a fixed eye, reasoning about the feasibility of a task simply consists of testing for the existence of a non-empty goal set, TGS. When the hand/eye system is mobile, as when the robot and the camera are mounted on a mobile platform with their relative positions fixed, reasoning about the goal should also result in generating a control sequence to move the platform, such that TGS becomes non-empty. In fact, movement of the observer can not only make a goal feasible but can also result in improving the resulting path. For example, in Figure 6, the goal set (TGS1) is changed to a new set (TGS2) that is easier to reach from the starting position of the hand. This transformation can be achieved simply by translation of the platform, for example, that also results in the translation of the PCS without changing its shape. This gives added flexibility since a better trajectory may be achieved. A extreme example is when the movement of the hand-eye system with the hand in a fixed state suffices for the dynamic task without any further movement of the hand. In terms of Figure 6, this would correspond to the situation when I0 comes to lie on TGS2, as a result of the movement of the platform. This is analogous to a baseball player moving, with the position of his hand relative to his eye more or less fixed, into the path of the ball, with perhaps a small corrective action just before catching the ball.

6. Vision-Based Control

The perceptual control surface can be divided into regular points, where the surface can be locally approximated by its tangent hyperplane (Figure 7a), facilitating a particularly simple control strategy using a steepest descent using the cost function $\mathcal{D}$ and singular points, where the Jacobian of π is zero. In its regular control mode, the robot monitors the sign of the Jacobian and can thus detect that it has crossed a singularity. Singularities of the control surface will typically be folds ([6]), which are degenerate along one direction of the control surface. The direction of a fold, which is defined by the kernel of the Jacobian matrix, can be directly determined from the measurements made, and

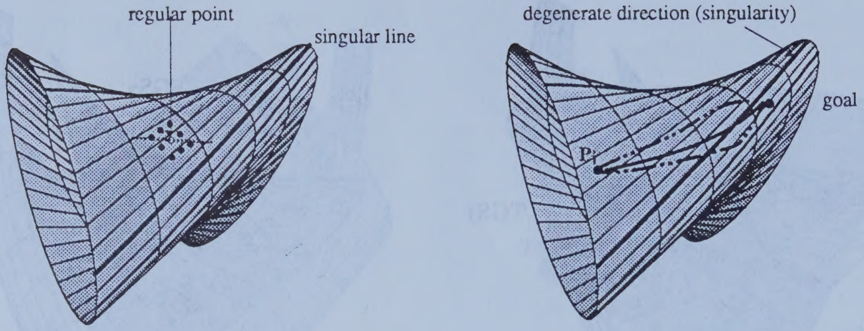


Figure 7: (a) Regular control by continuous visual feedback. (b) Dealing with singularities.

qualitatively identifies the singularity. In the planning stage, an attempt would be made to plan a path either around the singularities (if possible) or directly across them. In Figure 7b, the dashed line paths would be preferred over the path marked as solid line because the singular line would be crossed at a greater angle, minimizing the possibility of error caused by staying in the vicinity of a singularity. In order to follow the trajectory planning strategies discussed before, we need to know at each step the current position of the hand in the image and a reasonable estimate of the Jacobian matrix of the PKM at $\mathbf{q}$, $\mathbf{J}_m(\mathbf{q})$, which is improved as the robot moves. However, we can only compute the directional derivative (in the displacement direction, $\mathbf{u}$):

$$D_u\pi(\mathbf{q})\mathbf{J}_m(\mathbf{q})\cdot\mathbf{u}.$$

The tracking of the hand features and the updating of the Jacobian matrix (with partial information) is done using a Kalman filter ([2]). The gain terms for the Jacobian are changed adaptively, depending on the "distance" to the goal.

7. Preliminary Experiments

We have successfully demonstrated the feasibility of our approach by implementing the individual elements of the hand/eye system on available hardware—a 5-DOF Mitsubishi Movemaster II manipulator, a CCD camera, and a MacIntosh II for processing and control. In particular, we have developed a module for the visual tracking of the hand features using a Kalman filter. Although neither the camera nor the manipulator was calibrated—their relative position and orientation were unknown—our experiments indicated that the trajectories of the parameters we chose agreed closely with the predicted ones, and the mapping (PKM) was quite smooth. At this stage of the project, the image processing was purposely kept simple: a white rectangle was inscribed on the hand in order to facilitate the recovery of the image parameters. In order to further explore the possibilities and limitations of the PKM-based control, we have also written a software simulator for the classical 6-DOF PUMA 560 series robot. The details of the experimental work will be reported elsewhere.

8. Discussion

One goal of our framework has been to fuse the three levels defined in Section 2., combining the individual visual coordination tasks and relieving the burden on the sensor-feedback mechanism by allowing for a more 'qualitative' task definition. At the same time we have a framework for learning characteristics of the hand/eye system that could be varying with time or which would be known only at the time of execution. A synergism between high level symbolic reasoning (which forms the main body of AI literature) and low level numeric methods (robot control, computational vision, *etc.*) is important in the development of a sensor-based real-time robot operating with a certain degree of autonomy. This is also exemplified by the few successes in experimental systems, such as the now classic, robot ping-pong player of Andersson ([1]).

To avoid actual dynamics computations, we consider fixed worst-case limits on joint position, velocity, and acceleration. In order to exploit the dynamic capability of a manipulator more fully, however, this assumption may have to be relaxed. However, since the PCS—which is based on the kinematics—would not change, we can still use our framework to incorporate the robot dynamics considerations.

Our single correction scheme is suited for simple (slowly changing) target trajectories. If the target moves more unpredictably then multiple correction or even continuous estimation schemes may be more effective (*e.g.* [5, 12]). Continuous feedback at the goal definition level will introduce the additional problem of synchronization between visual trajectory estimation and control (see [12]).

Although it may be done under the same framework, we have not addressed here the fine planning and control needed at the final stages of the manipulation task (see [13]), *e.g.* when a target is about to be grasped. Instead, we consider the gross, vision-based planning and control that may be applicable to a wider class of dynamic manipulation tasks.

Another aspect not considered here is the possibility of having obstacles in the workspace of the manipulator. In terms of the PCS, however, this would correspond to "gaps" on the surface making the trajectory planning more complicated.

A higher degree of mobility can be achieved if the camera is 'active' and can be dynamically repositioned with respect to the robot. Then, the effect of camera movements can be assimilated into perturbations that alter the topology of the perceptual control surface. This in turn can be used favorably for dynamic reshaping of the PCS to better define and execute the manipulation task. This is a topic of future research, though the general idea of having a completely integrated control system that also factors in the variable position of the camera was outlined in [8].

9. Conclusion

We have presented a study of the role of vision in the autonomous operation of a robot for a family of dynamic manipulation tasks. The suggested framework provides a robust vision-based control, the analysis of which is based on qualitative, topological features of a perceptual control surface. As a result, the requirement of precise calibration of the different parts of the system is relaxed. Besides, the vision computations are feasible in real-time since only the most relevant measurements are used for tracking the image features—as also verified by our preliminary experiments. The approach can be adapted to different dynamic hand-positioning tasks (e.g. by modifying the cost function $\mathcal{D}$) so that it has promise for 'intelligent' operation of a robot.

References

[1] R. L. Andersson. *A Robot Ping-Pong Player. Experiment in Real-Time Intelligent Control.* MIT Press, Cambridge, 1988.

[2] C. K. Chui and G. Chen. *Kalman Filtering with Real-Time Applications.* Springer-Verlag, 1987.

[3] J. H. Connell. A Behavior-Based Arm Controller. *IEEE Transactions on Robotics and Automation*, 5:784-791, 1989.

[4] P. Cucka, R. Sharma, and J-Y. Hervé. Grasping Goals for Visual Coordination. In *Proc. 1991 IEEE International Conference on Systems, Man, and Cybernetics,* October 1991.

[5] J. T. Feddema and O. R. Mitchell. Vision-Guided Servoing with Feature-Based Trajectory Generation. *IEEE Transactions on Robotics and Automation*, 5:691–700, 1989.

[6] M. Golubitsky and V. Guillemin. *Stable Mappings and Their Singularities.* Springer-Verlag, New York, 1973.

[7] J-Y. Hervé, R. Sharma, and P. Cucka. Robust Vision-Based Control: Hand/Eye Coordination Without Calibration. In *Proc. 1991 IEEE Symposium on Intelligent Control*, August 1991.

[8] J-Y. Hervé, R. Sharma, and P. Cucka. The Geometry of Visual Coordination. In *Proc. Ninth National Conference of AAAI*, pages 732-737, July 1991.

[9] J. M. Hollerbach. A Survey of Kinematic Calibration. In O. Khatib, J. J. Craig, and T. Lozano-Pérez, editors, *The Robotics Review 1.* MIT Press, 1989.

[10] B. W. Mel. *Connectionist Robot Motion Planning.* Academic Press, New York, 1990.

[11] W. T. Miller, III. Sensor-Based Control of Robotic Manipulators Using a General Learning Algorithm. *IEEE Journal on Robotics and Automation*, 3:157–165, 1987.

[12] A. J. Koivo and N. Houshangi. Real-Time Vision Feedback for Servoing Robotic Manipulator with Self-Tuning Controller. *IEEE Transactions on Systems, Man, and Cybernetics*, 21:134–142, 1991.

[13] P. Liang, J-F. Lee, and S. Hackwood. A General Framework for Robot Hand-Eye Coordination. In *Proc. IEEE International Conference on Robotics and Automation*, pages 1082–1087, 1988.

[14] Z. Lin, V. Zeman, and R. V. Patel. On-Line Robot Trajectory Planning for Catching a Moving Object. In *Proc. IEEE International Conference on Robotics and Automation*, pages 1726–1731, 1989.

[15] A. C. Sanderson and L. E. Weiss. Adaptive Visual Servo Control of Robots. In A. Pugh, editor, *Robot Vision.* IFS Pub., Bedford, UK, 1983.

[16] S. B. Skaar, W. H. Brockman, and R. Hanson. Camera-Space Manipulation. *International Journal of Robotics Research*, 6(4):20–32, 1987.

[17] S. B. Skaar, W. H. Brockman, and W. S. Jang. Three-Dimensional Camera Space Manipulation. *International Journal of Robotics Research*, 9(4):22–39, 1990.

[18] R. Sharma, J-Y. Hervé, and P. Cucka. A Framework for Vision-Guided Manipulation of a Moving Target. In *Proc. 1991 IEEE International Conference on Systems, Man, and Cybernetics*, October 1991.

[19] R. Y. Tsai. Synopsis of Recent Progress on Camera Calibration for 3D Machine Vision. In *The Robotics Review 1.* MIT Press, Cambridge, Mass., 1989.

[20] L. E. Weiss, A. C. Sanderson, and C. P. Neuman. Dynamic Sensor-Based Control of Robots with Visual Feedback. *IEEE Journal of Robotics and Automation*, 3:404–417, 1987.

Symbolic Organization of Continuous Control

R. James Firby
Computer Science Department
University of Chicago
1100 East 58th Street, Chicago, IL 60637
firby@cs.uchicago.edu

Abstract

This paper is about the interface between continuous and symbolic robot control. We advocate describing continuous actions and their related sensing strategies as *situation specific activities*, which can be manipulated by a symbolic reactive planner. The approach addresses the problem of turning symbolic actions into continuous activities, and the problem of configuring the processing of high-bandwidth input, such as digital images, to support those activities.

1 Introduction

Consider the problem of a general purpose mobile robot following a person through a crowded room. The robot will have to track the person and move smoothly through an area containing both other people and fixed obstacles like desks and chairs. The robot must speed up and slow down so as not to either hit or lose the person it is following and it must turn without hitting other objects or people moving about the room. Furthermore, if the robot looses track of the person it is following it must look around or even search the area to find that person again. The robot must use sensing and action in tight feedback loops to track and move smoothly but it must also be able to adopt completely different goals and behaviors when the task it is performing changes character significantly.

Controllers for continuous processes are particularly good at managing actions that rely on the use of real time feedback, involve smooth motion over extended time, or require the coordination of multiple concurrent behaviors. For example, following a person might include: tracking by continuously pointing a video camera at the person and processing the image to give the direction to move, using that direction as feedback to steer the robot toward the person, and using sonar to monitor for nearby obstacles that need to be avoided. On the other hand, controllers are not as strong at shifting between completely different behaviors when the situation demands. It may be easy to notice when the person being tracked is lost from the image but searching to find the person again requires completely reconfiguring the feedback loops controlling the robot. Control functions have an even more difficult job if the robot also has the option of moving on to another task after a fruitless search.

Symbolic planning is especially good at achieving different goals at different times, coordinating multiple goals, and building sequences of discrete subtasks for achieving those goals effectively. For example, when the robot looses track of the person it is following, a planner might stop the robot and consider alternative courses of action. It might try and signal the person if the situation warrants, perhaps by making a beeping sound; it might decide to wait for the person to return for the robot later; or it might embark on a systematic strategy for searching the room in an attempt to find the person again. When choosing among these alternatives, the planner draws on knowledge of the situation (perhaps a beep is improper because the robot is in a library), knowledge about the likely behavior of the person being followed, and even the relative importance of finding the person versus moving on to some other goal. However, symbolic planning performs poorly at moving and coordinating continuous processes in situations where feedback is necessary.

A real robot will need both planners and controllers.

2 Background

Traditional AI work on robot control has focussed on the symbolic planning aspects of the problem [4, 8]. A robot is assumed to have a detailed model of the situation it starts in and a complete list of the goals it must achieve. The robot is also assumed capable of performing some variety of well characterized "primitive actions". To control the robot, the planner generates a sequence of primitive actions that are predicted to move the robot from its starting situation to a situation in which all of its goals have been achieved. In such a system, the process of following someone would typically be considered a single primitive action.

2.1 Reactive Planning

The shortcomings of the classical planning approach to robot control are now becoming well known [5, 6]. In particular, the robot will almost never have enough information to predict the details of the situations it will find itself in as it progresses through its plan. As a result, the robot will continually be encountering situations different from those it has planned for and the best sequence of primitives for achieving its goals will have to be recalculated again and again.

These difficulties have led to the idea of reactive planning [7, 12, 9]. A reactive planning system leaves as many primitive action choices as possible until run time when the robot will be able to determine the actual situation via direct sensing. Delaying commitment means there is little time to build a plan so reactive planners employ various means for caching planning decisions for fast lookup. Firby's [7] RAP reactive execution system maintains a hierarchy of plan fragments that can be expanded quickly as the situation unfolds and Schopper's system of universal plans creates decision trees that are searched before each action is executed [12].

2.2 Behavioral Control

Recently, to avoid the difficulties of caching plans and having to reconsider courses of action when information is uncertain, some AI researchers have argued for the abandonment of planning altogether and the use of only continuous control techniques. The most successful examples of such systems in the AI literature are organized around the idea of behaviors. Rather than generate a single control function that maps all inputs into all outputs, the system is broken down into relatively independent processes that monitor a subset of inputs and produce a specific result. The results are then combined to drive the outputs in a coherent fashion. This modularization gives the control system more structure and makes it easier to build and understand [11].

Behavior-based control systems have been very successful on quite sophisticated robots. Brooks has made particular progress using his subsumption architecture [3]. Using subsumption, each behavior is described by a finite automaton and the overall behavior of the system arises from the interaction of the automata with the environment and each other. Typical behaviors in a mobile robot might be "move away from obstacles", "move forward", and "turn towards the light". The more complex task of "move toward a light without hitting anything" is achieved by running these simple behaviors together. Horswill has used this architecture to build a robot which follows arbitrary moving objects [10]. Soldo uses a similar notion of interacting simple behaviors to navigate robustly in office hallways [13].

2.3 Flexible Control Systems

Symbolic planners rely on well defined primitive actions that can be combined in rich and flexible ways. Each primitive action is expected to be a self-contained control program and the planner is responsible for making sure that the external situation allows the action to execute properly when it is used. Unfortunately, when the world is very complex, and the planner can distinguish many different situations, there must be a huge number of primitive actions. A different action is needed for following a person, for traveling down a hallway to the end, for moving down a hallway to the second door on the right, and so on. This explosion of primitive actions either limits the symbolic approach to restricted worlds or requires a huge effort to program the underlying control system.

Behavioral control systems rely on a single control program rather than a library of primitive actions. The idea is to build much more responsiveness to the environment into the control program than could easily be managed by a symbolic planner. However, the trade-off is that all of the goals, and combinations of goals, that the system can carry out must be programmed in advance. In a complex world, building all goals into the robot ahead of time limits it to simple sets of tasks.

In both of these cases, the robot is limited to simple domains because the control system of the robot must be programmed once and for all. Either the control system must contain programs for a multitude of primitive actions or it must contain a single program that incorporates all possible goals. Planning systems are supposed to circumvent this problem by, in effect, reprogramming themselves each time a new goal is introduced or the situation changes significantly. They strive to be *situation specific*, tailoring their plans

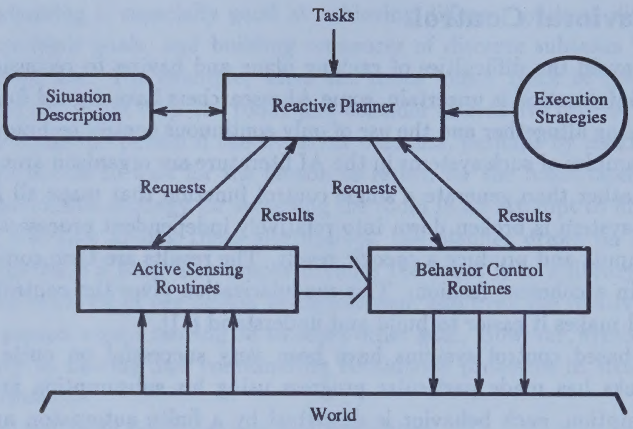


Figure 1: An Integrated Agent Architecture

to exactly what is happening. However, situation specific plans bottom out in situation specific primitive actions and when the control system cannot be reprogrammed those primitives must all be developed in advance.

Our goal is to construct a control system that exhibits the advantages of situation specific behavior and can be used to implement multiple, varied and changing plans. Our approach is to build a behavior-based control system that can be "reprogrammed" by a symbolic reactive planner overseeing its operation. Each configuration of the control system will have the moment to moment characteristics of continuous control but will have emergent discrete properties over longer timescales so that planners can treat a configuration like a primitive action.

3 Situation Specific Robot Control

The system we are building to combine symbolic planning with continuous control is illustrated in Figure 1. The architecture consists of three main components: a reactive planner, a collection of active sensing routines, and a collection of behavioral control routines. The reactive planner is based on the RAP system described in [6, 7]. It manages the robot's goals, a description of the current situation, and a library of execution strategies that descibe methods for carrying out goals in different situations. The RAP system uses the situation description to choose execution strategies for the robot's goals at run time and reconsiders those choices as the situation changes so that it can react quickly and change plans when the situation warrants.

The active sensing and behavioral control routines form the heart of the real-time control system. Drawing on the work in behavioral control described in Section 2.2 the control system is split into relatively independent sensing and action routines that can be activated, deactivated, and reconfigured individually. By carefully choosing the order in which routines are enabled and disabled, the planner can produce behavior sets with stable properties that extend over a period of time. For example, enabling collision avoidance behaviors, a visual tracking routine, and a routine to move the robot in the direction of the tracked feature, the planner can produce a state in which the robot follows a person without bumping into anything. This state will last indefinitely without further intervention as long as the tracked feature stays in view and the path toward it stays reasonably clear.

By shifting between stable sets of active routines, the planner breaks the continuous world of control into discrete activities designed to achieve a particular goal. These activities are similar to primitive actions in the way they can be treated as discrete entities and used as plan building blocks to produce a predictable action. However, activities differ from atomic actions because they can be created dynamically and adapted to changing situations at run time.

The ability to produce and use *situation specific activities* is the corner stone of our architecture. The rest of this paper discusses the issues involved in building stable activities that can be treated like discrete actions by the reactive planner.

4 Situation Specific Activities

The goal in defining a situation specific activity is to construct a configuration for the control system that predictably achieves a particular goal in the real world. In our system, these configurations (and hence activities) will be constructed by the reactive planner as requests to enable, disable, and parameterize routines in the control system. Once activated, the routines will run continuously as feedback control loops without intervention. The problem is to define configurations that *reliably* move the robot through a task in an uncertain, dynamic environment.

Sometimes defining an activity is easy because the desired goal requires only a single routine. Taking a sonar range reading or pointing the camera in a particular direction will typically require invoking a single hardware action. In such cases, the goal of the action is directly embodied in the arrangement of the hardware being used.

Usually, however, the activities useful in plans will be much more complex than a single hardware invocation and will require a whole set of routines for their definition. The person following example requires a routine to move the robot in the tracked direction, another to direct the robot away from obstacles, and yet another to keep the robot from getting to close or too far away from the person. Executing "follow person" as a single action means invoking all of these routines together. Once activated, their interaction with each other and the world will have the *emergent* property of satisfying the desired goal. Moreover, during the time the robot is following the person, the control system doesn't change — the same routines are running doing the same thing all of the time. A short term stable situation has been generated. From the robot's point of view, a single control state persists over an interval of time and a discrete action has been carved out of the continuous world.

We define a situation specific activity as a collection of control routines that work together over an interval of time to achieve a clear symbolic goal.

4.1 Guarded Situation Specific Activities

Situation specific activities are not enough to define useful actions by themselves. A critical feature of planned actions is that they are discrete and finite. Each action must be implemented as a stable control state with a well-defined beginning and end. When a situation specific activity is used to define an action, its beginning is clear and occurs when the activity is invoked. However, all of the routines in an activity will run happily in any situation and thus will run happily forever.

To be useful, activities must be finite; they must signal when they have successfully completed or when they discern that they can no longer achieve their intended goal. When the activities involve hardware actions with well defined completions this requirement is fulfilled simply by having the hardware signal when the operations are complete. The end of a sonar reading or a camera motion is unambiguous and clear.

However, when the purpose of an activity is emergent, there is no clear sign that it has been achieved. Moving forward, avoiding obstacles, and staying a constant distance from the wall might have the effect of moving the robot down a hall, but when is that action complete? Completion may mean many things: having traveled a specific distance, having reached the end of the hall, or having passed three doorways. Regardless of which of these goals is desired, noticing that it has been achieved requires monitoring states in the world beyond those needed just to carry it out. These additional monitoring activities must be included in any situation specific activity that implements a discrete action.

The activities that define actions must not just signal completion; they must also signal reliably when they fail. The real world is filled with uncertainty and real sensors are not 100% reliable. As a result, configurations of routines will not always achieve the goals that they were intended to achieve. This is particularly true when the desired goal is emergent. The routines making up such an activity may continue to execute happily yet no signal of success will occur. For example, suppose a robot is executing a set of routines designed to move it down a hall 10 ft. What happens if the hallway is only 5 ft. long? Presumably it will eventually come to a halt under the direction of its various obstacle avoidance routines but its completion monitor will not yet be satisfied. At that point something should happen or the robot will just sit there forever, secure in the fact that its active routines will move it reliably down the hall, not realizing that they have been stymied by the context the robot unwittingly finds itself in.

The solution to this problem is to augment the routines implementing the action to include monitor routines for failure states as well as for success states. This procedure is mandatory for every activity that is to define a reliable symbolic action. In the hallway, this may involve something simple like signaling that the robot has been stationary for 10 seconds, or something more complicated like signaling that the robot moving back and forth to go around the obstacle making up the end of the hall but is making no measurable progress. The important point is that the activity must *always* signal that it is wedged. Only with such assurances can the planning system trust that its planned actions will interact with the world in a sensible fashion.

We will define a guarded situation specific activity to be a set of control routines that together achieve a particular goal and always signal when the goal has been achieved, or when it has become impossible. As the symbolic system knows which routines it has activated to signal success, and which it has activated to single failure, it can tell whether an activity has completed normally or been interrupted simply by knowing which routine is signaling. There is no need for the control system to understand the implication or purpose of its constituent routines in order for the symbol system to understand what has occurred.

5 Situation Specific Sensing

Developing guarded actions is only half the story in defining realizable activities. Authors of control systems for real robots routinely make the claim that their systems work because sensing is restricted to precisely those features of the world that are relevant to the task at hand [10, 13]. Underlying this claim is the realization that processing all available sensor data is both too slow to support control and unnecessary in most cases. Sensing strategies can and should be specialized to the particular needs of the control routines that use them. Building a world model, or matching sensor readings against an existing world model, is typically not a bounded time procedure and would endanger response time.

Some types of sensor data are inherently specialized and require little interpretation. Touch sensors might read `pressed` or `not-pressed` and sonars might generate direct range readings. The interpretation of such data is easy because it does not rely on context. The meaning of the data is defined by the sensor and its construction and is independent of the circumstances in which the agent finds itself. Buttons and sonars remain the same no matter what the planner's goals or situation become.[1]

However, the interpretation of many other forms of sensor data does depend on context. In particular, computer vision algorithms have typically required a significant application of previous knowledge, expectations about what is being viewed, and complex processing directed by this knowledge and these expectations. This would suggest that vision cannot be used as a reliable sensor in control systems. However, recent research in *active vision* has suggested that, in fact, it can.

5.1 Active Vision

Active vision algorithms are designed specifically to be part of situation specific activities [2]. They can take advantage of the constraints provided by the motion of the observer, and are designed to extract precisely the knowledge that is required for an activity to function successfully, within the time constraints required by the agent and its environment.

An active vision algorithm may be restricted in many ways to conserve computational resources. It may only examine a small region of the world potentially visible in angle and in depth. It will be restricted in its purpose, depending on the role it is fulfilling within the task the system is carrying out. And it will use the most pertinent cues that best fulfill the role given what is known about the situation at hand.

---

[1] Anyone who has played with real sonars knows that this isn't quite true. The meaning that should be attached to a sonar range reading often depends critically on the object producing the reflection. However, raw range data is always available.

For example, consider the ways active adjustment of viewing parameters can help solve visual problems. Maintaining gaze on an object stabilizes the object on the photoreceptor array, aiding motion interpretation [1]. Control of zoom can bring an object into higher resolution for more careful examination, or allow a greater area to be examined when searching for an object. Zoom control can be combined with camera movement strategies to get the object within view at a high enough resolution to be recognized [15].

Like buttons and sonars which generate data that can be used without complex interpretation in context, the data generated by active vision routines can also be used directly. For example, the center of an object being tracked can be used as a position for the robot to travel towards. The resulting emergent behavior has the robot follow the object even though following is not explicitly represented and neither recognition nor interpretation of the object is required.

5.2 Extracting Functional Attributes

Attributes of an object like its center, direction, or speed are important because they are useful in many different situations for accomplishing many different tasks. For example, the direction to an object can be used by routines designed to approach the object, to avoid the object, and to point in the object's direction. As long as the direction is known, these activities can proceed. It is the direction attribute that has a functional use in the routines, not the object itself or the way that the attribute is extracted.

Functional attributes are the features of the world that need to be computed in real time for use in feedback loops. Each attribute that a robot can extract, corresponds to a particular interpretation of its sensor data that is relevant to some activity that it might pursue. When behavioral control researchers suggest adding a new sensor when a new behavior is to be encoded, what they really mean is to add a new real-time functional attribute extractor. Active vision routines are a particularly useful idea in this respect because they are designed to extract different attributes from the world using the same sensor.

In fact, there will often be many different active vision routines for extracting the same functional attribute from the world. This allows a planner substantial leeway to instantiate its activities in different ways under different circumstances using different active vision routines. For example, tracking the location of an obstacle might be done using color, shape, or motion, depending on what is most reliable in the specific context. Should a planner want to go around an obstacle, it must choose both a guarded set of routines to carry out the motions and a tracking strategy to use for extracting the object's position. When selecting a tracking strategy, the planner uses its knowledge of the obstacle to focus the interpretation of visual data (*i.e.*, color, motion, shape) down to just that which is necessary to reliably keep track of the object's relevant functional attribute (*i.e.*, position).

Guarded situation specific activities define emergent symbolic actions that can be carried out in the world. Active vision routines compute the functional attributes of items in the world that are used within activities. Thus, if we think of an activity as describing a symbolic action, we can think of active vision routines as describing the arguments to that action.

6 Conclusion

We are attempting to combine reactive execution and continuous control within a single architecture that employs the strengths of each to overcome the shortcomings of the the other. The central idea within the architecture is that of an *activity.* Activities are essentially control programs assembled at run time using instructions from the planning system to active sensing and behavioral control routines. By reprogramming the control system at run time, a small control system can be used to generate a large number of different, discrete actions.

Our architecture draws on the following insights:

- *Control systems can be built up from specialized routines that act on only a small portion of available data from the world.*
  By tuning a system's sensing and action routines to exactly the task at hand, very complex interaction with the world can be achieved using a relatively simple control system. It is only necessary to process and act on that information from the world that is relevant to the task at hand.
- *The actions required to achieve a goal can be chosen, and hence specialized, at run time when the details of the situation they will be used in is known.*
  By leaving action choices (and hence activity building) until run time, a reactive planner can tailor its actions to the precise situations encountered. Actions become both goal and context directed.

An initial version of our architecture is currently being implemented on a robot at the University of Chicago to handle the person tracking task discussed through this paper. It includes two different visual tracking routines that are useful in different situations. A color histogram tracker [14] can be used to track an object with a specific color signature and an image difference tracker can be used to track moving objects that do not have a distinct color signature. Behavior control routines have been implemented to move in a given direction and to use sonar and infra-red range data to avoid collisions. At present, the robot can track a moving person, avoid collision and initiate a visual search if the person moves too rapidly out of view

We will continue to expand the vocabulary of sensing and action routines and the tasks they can be used with. We hope the result will be a generally useful, robot independent way for describing ordinary, everyday behaviors.

References

[1] Yiannis Aloimonos, Isaac Weiss, and Amit Bandyopadhay. Active vision. In *Proceedings of the Image Understanding Workshop*, pages 552–573, Los Angeles, California, February, 1987.

[2] Dana H. Ballard. Reference frames for animate vision. In *International Joint Conference on Artificial Intelligence*, pages 1635–1641, 1989.

[3] Rodney A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA-2(1), March 1986.

[4] David Chapman. Planning for conjunctive goals. Technical Report TR 802, MIT Artificial Intelligence Laboratory, 1985.

[5] David Chapman and Philip E. Agre. Abstract reasoning as emergent from concrete activity. In *Workshop on Planning and Reasoning about Action*, Portland, Oregon, June 1986.

[6] R. James Firby. An investigation into reactive planning in complex domains. In *Sixth National Conference on Artificial Intelligence*, Seattle, WA, July 1987. AAAI.

[7] R. James Firby. Adaptive execution in complex dynamic worlds. Technical Report YALEU/CSD/RR #672, Computer Science Department, Yale University, January 1989.

[8] R. James Firby and Drew McDermott. Representing and solving temporal planning problems. In N. Cercone and G. McCalla, editors, *The Knowledge Frontier*. Springer Verlag, New York, NY, 1987.

[9] Michael P. Georgeff, Amy L. Lansky, and Marcel J. Schoppers. Reasoning and planning in dynamic domains: An experiment with a mobile robot. Tech Note 380, AI Center, SRI International, 1986.

[10] Ian Douglas Horswill and Rodney Allen Brooks. Situated vision in a dynamic world: Chasing objects. In *Seventh National Conference on Artificial Intelligence*, St. Paul, MN, August 1988. AAAI.

[11] David Payton. Exploiting plans as resources for action. In *Workshop on Innovative Approaches to Planning, Scheduling, and Control*, San Diego, CA, November 1990. DARPA.

[12] M.J. Schoppers. Universal plans for reactive robots in unpredictable environments. In *Tenth International Joint Conference on Artificial Intelligence*, Milan, Italy, August 1987. IJCAI.

[13] Monnett Hanvey Soldo. Fusion without representation. In *Vol. 1198 Sensor Fusion II: Human and Machine Strategies*, Philadelphia, PA, November 1989. SPIE.

[14] Michael J. Swain. Color indexing. Technical Report 360, Department of Computer Science, University of Rochester, 1990.

[15] Lambert E. Wixson. Object search: A pervasive (but unstudied) visual task. Technical report, University of Rochester Computer Science Department, February 1990. in preparation.

Scalability of Parallel Algorithms and Architectures*

Vipin Kumar and Anshul Gupta

Department of Computer Science
University of Minnesota
Minneapolis, MN - 55455
(612) 625-4002
kumar@cs.umn.edu and *agupta@cs.umn.edu*

Abstract

The scalability of a parallel algorithm on a parallel architecture is a measure of its capability to effectively utilize an increasing number of processors. The scalability analysis of a parallel algorithm and a parallel architecture combination may be used to predict its performance for a large number of processors from the known performance on fewer processors. For a fixed problem size it may be used to determine the optimal number of processors to be used and the maximum possible speedup that can be obtained. In this paper we assess the state of the art in scalability analysis, and motivate further research on the development of new and more comprehensive analytical tools to study the scalability of parallel algorithms and architectures. We discuss the interrelationships between a number of techniques and formalisms that have been developed for studying the scalability issues. We also address issues pertaining to cost scalability and the impact of technology dependent factors on scalability.

1 Introduction

Parallel computers employing hundreds to thousands of processors are commercially available today, and in future, much bigger parallel systems are expected to be available. Availability of such systems has made it important to investigate the performance of parallel computers containing a large number of processors. The scalability of a parallel algorithm on a parallel architecture is a measure of its capability to effectively utilize an increasing number of processors. It is important to perform scalability analysis as one can reach very misleading conclusions regarding the performance of a large parallel system if one attempts to simply extrapolate its performance based on that of a similar smaller system. Scalability analysis of a parallel algorithm-architecture combination can be used for a variety of purposes. It may be used to predict the performance of a parallel algorithm and a parallel architecture for a large number of processors from the known performance on fewer processors. It may be used to select the best parallel algorithm-architecture combination for a problem. For a fixed problem instance it may be used to determine the optimal number of processors to be used and the maximum possible speedup that can be obtained. The scalability analysis can also predict the impact of changing hardware technology on the performance and thus help design better parallel architectures for solving various problems. Hence development of tools for scalability analysis is currently an active area of research.

---

*This work was supported by Army Research Office grant # 28408-MA-SDI to the University of Minnesota and by the Army High Performance Computing Research Center at the University of Minnesota. This paper is based upon references [20] and [21].

The objective of this paper is to assess the state of the art in scalability analysis, and to motivate further research on the development of new and more comprehensive analytical tools to study the scalability of parallel algorithms and architectures. In this paper we discuss the interrelationships between a number of techniques and formalisms that have been developed for studying the scalability issues. We also address issues pertaining to cost scalability and the impact of technology dependent factors on scalability.

2 The Isoefficiency Metric For Scalability

We have developed a scalability metric, called the *isoefficiency function*, which relates the size of the problem to be to solved the number of processors for an increase in speedup in proportion to the number of processors used [22, 23]. An important feature of isoefficiency analysis is that in a single expression, it succinctly captures the effects of characteristics of the parallel algorithm as well as the parallel architecture on which it is implemented. By performing isoefficiency analysis, one can test the performance of a parallel program on a few processors, and then predict its performance on a larger number of processors. Isoefficiency analysis has been found to be very useful in characterizing the scalability of a variety of parallel algorithm-architecture combinations [10, 12, 20, 22, 23, 24, 25, 32].

Before formally defining the notion of isoefficiency, we give a few definitions and assumptions. A parallel computer consists of an ensemble of p processing units each of which runs at the same speed. In the execution of a parallel algorithm, the time spent by processor P_i can be split into t_e^i, the time spent in useful computation; and t_o^i, the time spent in doing communication, idling and other work which would not have been performed by the optimal or the best known sequential algorithm but is necessitated due to parallel processing. The Problem Size W is defined to be the total amount of computation done by the optimal or the best known sequential algorithm and is a function of the input data size n. Clearly $W = \Sigma_i\ t_e^i$. For example, in an FFT computation over n input data elements, the problem size W is $\Theta(n \log n)$. We define the total overhead T_o to be $\Sigma_i\ t_o^i$. The execution time on p processors, T_P, satisfies $T_P = t_e^i + t_o^i$ (for any i). Also we have $W + T_o = pT_P$.

Speedup S is the ratio $\frac{W}{T_P}$. Efficiency is the speedup divided by p. Therefore,

$$E = \frac{S}{p} = \frac{W}{T_P \times p} = \frac{W}{W + T_o} = \frac{1}{1 + \frac{T_o}{W}}$$

If a parallel algorithm is used to solve a problem instance of a fixed size (*i.e.*, fixed W), then the efficiency decreases as p increases. The reason is that T_o increases with p. For many parallel algorithms, for a fixed p, if the problem size W is increased, then the efficiency becomes higher (and approaches 1 or some upper-bound e between 0 and 1), because T_o grows slower than W. For these parallel algorithms, the efficiency can be maintained at a desired value (between 0 and e) for increasing p, provided W is also increased. We call such algorithms **scalable** parallel algorithms.

Note that for a given parallel algorithm, for different parallel architectures, W may have to increase at different rates w.r.t. p in order to maintain a fixed efficiency. The rate at which W is required to grow w.r.t. p to keep the efficiency fixed is what essentially determines the degree of scalability of the parallel algorithm for a specific architecture. For example, if W is required to grow exponentially w.r.t. p, then the algorithm-architecture combination is poorly scalable. The reason is that in this case it would be difficult to obtain good speedups on the architecture for a large number of processors, unless the problem size being solved is enormously large. On the other hand, if W needs to grow only linearly w.r.t. p, then the algorithm-architecture combination is highly scalable and can easily deliver linearly increasing performance with increasing number of processors for reasonable problem sizes. Since all problems have a sequential component (at least equal to one arithmetic operation), W must asymptotically grow at least linearly w.r.t. p to maintain a given efficiency. If W needs to grow as $f_E(p)$ to maintain an efficiency E, then $f_E(p)$ is defined to be the **isoefficiency function** for efficiency E and the plot of $f_E(p)$ w.r.t. p is defined to be the **isoefficiency curve** for efficiency E.

Since $E = 1/(1 + \frac{T_o}{W})$, in order to maintain a constant efficiency, W should be proportional to T_o. In other words, the following equation must be satisfied :

$$W = KT_o \tag{1}$$

Here $K = \frac{E}{1-E}$ is a constant depending on the efficiency E to be maintained. Once W and T_o are known as functions of n and p, the isoefficiency function can often be determined by simple algebraic manipulations.

3 Scalability Study of Algorithms Using the Isoefficiency Metric

The scalability of a variety of algorithms has been studied using the isoefficiency metric. In [12], we analyze the scalability of the parallel Fast Fourier Transform algorithm on mesh and hypercube connected multicomputers. The scalability analysis of FFT on the hypercube architecture provides several important insights. On this architecture, parallel FFT algorithm can obtain linearly increasing speedup with respect to the number of processors with only a moderate increase in problem size. But there is a limit on the achievable efficiency and this limit is determined by the ratio of CPU speed and communication bandwidth of the hypercube channels. Efficiencies higher than this limit can be obtained only if the problem size is increased very rapidly. It is also shown that pipelining the communication, thereby overlapping it with the computation, does not improve the scalability significantly. The scalability analysis for the mesh connected multicomputers reveals that FFT cannot make efficient use of large-scale mesh architectures. It is shown that addition of features such as cut-through-routing (also known as worm-hole routing) to the mesh architecture do not improve the overall scalability characteristics of the FFT algorithm on this architecture. We also show that under certain assumptions, it is more cost-effective to implement the FFT algorithm on a hypercube rather than a mesh despite the fact that large scale meshes are cheaper to construct than large hypercubes. All the results in [12] hold for ordered and multi-dimensional FFT as well.

The scalability of a class of load balancing algorithms that are used in parallel formulation of tree search based applications [17] is analyzed in [22, 29, 23, 24]. In these problems, the work done by a processor can be partitioned into independent work pieces, which require either no or small amount of communication. But load balancing is difficult because the work pieces are of highly variable sizes and it is not possible (or very difficult) to estimate the size of total work at a given processor. Hence any load balancing scheme has to dynamically transfer work among different processors. We investigated the performance of many different schemes experimentally as well as analytically for a variety of architectures. Many of the previously known schemes performed well on a small number of processors (eg. 16 - 64) but performed very poorly when used on a large number of processors (128 - 1024). Our scalability analysis of these schemes motivated development of optimally scalable load balancing schemes for a variety of problems. These optimal schemes were incorporated in tree search programs for solving a variety of practical problems such as optimizing floor-plan of a VLSI chip [1], generating test patterns for combinatorial circuits [3] and tautology verification [2]. We were able to demonstrate linear speedup as the number of processors was increased upto 1024 processors (the largest configuration available) on Ncube/10 and upto 128 processors (the largest configuration available) on Intel Hypercube, Symult 2010 and BBN Butterfly.

In [25], the scalability of several parallel algorithms for finding shortest paths between all pairs of nodes in a densely connected graph on mesh, hypercube and shared-memory architectures is discussed. This analysis helps in determining the algorithm-architecture combinations that are superior to others. In [16] some predictions about comparative behavior of two of the algorithms were made on the basis of limited experimental results. These are shown to be incorrect by our scalability analysis.

In [32], the scalability of a variety of parallel sorting algorithms on the mesh multicomputers is analyzed. Many sorting algorithms are originally designed to work for the case in which there is one processor for each element. If there are fewer processors than the number of elements to be sorted, then the algorithm has to be adapted to work with more than one elements per processor. We show that algorithms derived by simply allowing a processor to multiplex as more than one processor (a technique advocated in the data-parallel programming paradigm [15] leads to poorly scalable parallel algorithms.

4 How does the Isoefficiency Metric Relate to Other Scalability Measures?

In Addition to the isoefficiency metric, a number of techniques and formalisms have been developed for studying scalability issues. After reviewing the various measures of scalability, one may ask whether there exists one scalability metric that is better than all others? The answer to this question is no, as different metrics are suitable for different situations.

One situation arises when the problem at hand is fixed and one is trying to use an increasing number of processors to solve it. In this case, the speedup is determined by the serial fraction in the program as well as other overheads such as those due to communication and due to redundant work. In this situation choosing one parallel system over the other can be done using the standard speedup metric. Note that for any fixed problem size W, the speedup on a parallel system will saturate or peak at some value $S^{max}(W)$, which can also be used as a metric. Scalability issues for the fixed problem size case are addressed in [9, 18, 11, 30, 34, 36].

Another possible scenario is one in which a parallel computer with a fixed number of processors is being used and the best parallel algorithm needs to be chosen for solving a particular problem. For a fixed p, the efficiency increases as the problem size is increased. The rate at which the efficiency increases and approaches 1 (or some other maximum value) with respect to increase in problem size may be used to characterize the quality of the algorithm's implementation on the given architecture.

The third situation arises when the additional computing power due to the use of more processors is to be used to solve bigger problems. Now the question is how should the problem size be increased with the number of processors?

For many problem domains, it is appropriate to increase the problem size with the number of processors so that the total parallel execution time remains fixed. An example is the domain of weather forecasting. In this domain, the size of the problem can be increased arbitrarily provided that the problem can be solved within a specified time. (It does not make sense to take more than 24 hours to forecast the next day's weather.) The scalability issues for such problems have been explored by Worley [36], Gustafson [14, 13], and Sun and Ni [33].

Another extreme in scaling the problem size is to try as big problems as can be handled in the memory. This is investigated by Worley [36, 37], Gustafson [14, 13] and by Sun and Ni [33], and is called the memory constrained case. Since the total memory of a parallel computer increases with increasing p, it is possible to solve bigger problems on parallel computer with bigger p. It should also be clear that any problem size for which the memory requirement exceeds the total available memory cannot be solved on the system.

An important scenario is that in which one is interested in making efficient use of the parallel system; *i.e.*, it is desired that the overall performance of the parallel system increases linearly with p. Clearly, this can be done only for scalable parallel systems, which are exactly the ones for which a fixed efficiency can be maintained for arbitrarily large p by simply increasing the problem size. For such systems, it is natural to use isoefficiency function or related metrics [23, 19, 6]. The analyses in [38, 39, 9, 27, 30, 34, 8] also attempt to study the behavior of a parallel system with some concern for overall efficiency.

Although these different scalability measures are appropriate for rather different situations, many of them are related to each other. For example, from the isoefficiency analysis, one can reach a number of conclusions regarding the "fixed-time" case (i.e., when bigger problems are solved on larger parallel computers with some upper-bound on the parallel execution time). It has been shown in [21] that the problem size can be increased linearly with the number of processors while still maintaining a fixed execution time if and only if the isoefficiency function is $O(p)$.

A direct corollary of the above result is that if the isoefficiency is higher than linear, then the minimum parallel execution time will increase even if the problem size is increased as slowly as linearly with the number of processors. Worley [36, 37] has shown that for many algorithms used in the scientific domain, for any given T_P, there will exist a problem size large enough so that it can not be solved in time T_P, no matter how many processors are used. Our above analysis shows that for these parallel systems, the isoefficiency curve has to be worse than linear.

If the isoefficiency function of a parallel system is greater than $O(p)$, then given a problem size W, there is as lower-bound on the parallel execution time. In [11] we show that for a fairly wide class of overhead functions, the relation between problem size W and the number of processors p at which the parallel run time T_P is minimized, is given by the isoefficiency function for some value of efficiency. For these parallel systems, no advantage (in terms of parallel execution time) is gained by making inefficient use of the parallel processors.

5 Analyzing the Impact of Technology Dependent Factors on Scalability

The changes in technology dependent factors such as computation speed and communication speed can have a significant effect on the performance of a parallel system. It is clear that if the CPU or the communication speed is increased, the overall performance can only become better. But unlike a sequential computer, in a parallel system, a k-fold increase in the CPU speed may not necessarily result in a k-fold reduction in execution time for the same problem size. A number of researchers have considered the impact of the changes in CPU and communication speeds on the performance of a parallel system [30, 26, 12, 25].

For instance, consider the implementation of the FFT algorithm on a SIMD hypercube [12]. The efficiency of a N point FFT computation on a p-processor cube is given by $E = \frac{1}{1+\frac{t_w}{t_c}\frac{\log p}{\log N}}$. Here t_w is the time required to transfer one word of data between two directly connected processors and t_c is the time required to perform a unit FFT computation. In order to maintain an efficiency of E, the isoefficiency function of this parallel system is given by $\frac{E}{1-E}t_w p^{\frac{E}{1-E}\frac{t_w}{t_c}} \log p$. Clearly, the scalability of the system deteriorates very rapidly with the increase in the value of $\frac{E}{1-E}\frac{t_w}{t_c}$. For a given E, this can happen either if the computation speed of the processors is increased (*i.e.*, t_c is reduced) or the communication speed is decreased (*i.e.*, t_w is increased). For example, if $t_w = t_c = 1$, the isoefficiency function for maintaining an efficiency of 0.5 is $p \log p$. If p is increased 10 times, then the problem size needs to grow $10\sqrt{10}$ times to maintain the same efficiency. On the other hand, a 10-fold increase in t_w or a 10-fold decrease in t_c will require one to solve a problem of size nearly equal to W^{10} if W was the problem size required to get the same efficiency on the original machine. In [12] it is also shown that no matter how fast the processors are, the throughput (in terms of number of FFT computations per unit time) of a p processor SIMD hypercube for an N-point FFT computation is upper-bounded by $\frac{p \log N}{t_w \log p}$. It is then shown that this upper-bound can be improved by either increasing the problem size exponentially or by improving the communication related parameters linearly. Thus the technology dependent constants like t_c and t_w can have a dramatic impact on the scalability of a parallel system.

In [32] and [10] respectively, the impact of these parameters on the scalability of parallel shortest path algorithms and the matrix multiplication algorithms is discussed. In these algorithms, the isoefficiency function does not grow exponentially (as in the case of FFT) with respect to the $\frac{t_w}{t_c}$ ratio, but is a polynomial function of this ratio. For example, in most parallel implementations of matrix multiplication on mesh architecture [10], the isoefficiency function is proportional to $(\frac{t_w}{t_c})^3$. The implication of this is that for the same communication speed, using ten times faster processors in the parallel computer will require a 1000 fold increase in the problem size to maintain the same efficiency. On the other hand, if p is increased by a factor of 10, then the same efficiency can be obtained on $10\sqrt{10}$ times bigger problems. Hence for parallel matrix multiplication, it is better to have a parallel processor with k-fold as many processors rather than a parallel processor with the same number of processors, each k-fold as fast (assuming that the communication network and the bandwidth etc. remain the same).

The above examples show that improving the technology in only one direction (*i.e.*, towards faster CPUs) may not be a wise idea. In other words, increasing the speed of the processors alone without improving the communicationspeed will result in diminishing returns in terms of overall speedup and efficiency. The decrease in performance is highly problem dependent. For some problems it is dramatic (*e.g.*, parallel FFT [12]),while for others (*e.g.*, parallel matrix multiplication algorithm [10]) it is less severe. In addition to CPU speed and communication bandwidth, there are many other technology dependent parameters that effect the performance and hence the scalability of parallel systems. Examples are the amount and speed of cache and main memory, vector processing capability, message start-up time, packet vs. circuit switching etc.

The tremendous impact that the technology dependent factors have on parallel performance makes it very important to perform scalability and performance analysis from the point of view of these factors. Such analysis will not only aid in performance prediction, but will also help in designing better and efficient parallel machines for solving classes of problems.

6 Study of Cost Effectiveness of Parallel Systems

So far we have confined ourselves to the study of scalability of a parallel system to investigating the capability of the parallel algorithm to effectively utilize an increasing number of processors on the parallel architecture. Many algorithms may be more scalable on more costly architectures (*i.e.*, architectures that are cost-wise less scalable due to high cost of expanding the architecture by each additional processor). For example, the cost of a parallel computer with p processors organized in a mesh configuration is proportional to $4p$, whereas a hypercube configuration of the same size will have a cost proportional to $p \log p$ because a p processor mesh has $4p$ communication links while a p processor hypercube has $p \log p$ communication links[1]. In such situations, one needs to consider whether it is better to have a larger parallel computer of a cost-wise more scalable architecture that is underutilized (because of poor efficiency), or a smaller parallel computer of a cost-wise less scalable architecture that is better utilized. For a given cost, the aim is to maximize the overall performance. Using a simple model of cost of Mesh and Hypercube multicomputers, it is shown in [12] that for parallel FFT, hypercube is more cost effective than the mesh even though a hypercube is costlier to build than a mesh with the same number of processors. On the other hand, it is quite possible that the implementation of a different parallel algorithm is more cost effective on an architecture on which the algorithm is less scalable. Thus mere scalability analysis is not adequate to choose the best architecture for a problem and cost analysis can be very valuable in determining the most suitable architecture for a class of applications.

Another issue in the cost vs. performance analysis of parallel systems is determining the tradeoffs between the speed of processors and the number of processors to be employed for a given budget. From the analysis by Barton and Withers in [4] and by Van-Catledge in [35], it may appear that higher performance is always obtained by using fewer and faster processors. It can be shown that this is true only for those parallel systems in which the overall overhead T_o grows faster than or equal to $O(p)$. The model considered in [35] is one for which $T_o = O(p)$ as they consider the case of a constant serial component.

---

[1] Here it is assumed that the cost is proportional to the number of links and is independent of the length of the links. A detailed analysis of cost issues in different multicomputers is done in [7].

As an example of a parallel system where their conclusion is not applicable, consider the matrix multiplication on a SIMD mesh architecture discussed in Section 2.1. Here $T_o = t_w N^2 \sqrt{p}$ for multiplying $N \times N$ matrices. Thus $T_P = t_c \frac{N^3}{p} + t_w \frac{N^2}{\sqrt{p}}$. Clearly, a better speedup can be obtained by increasing p by a factor of k (until p reaches N^2) rather than reducing t_c by the same factor.

Even for the case in which T_o grows faster than or equal to $O(p)$, it might not always be cost effective to have fewer faster processors. The reason is that the cost of a processor increases very rapidly with respect to speed once a technology dependent threshold is reached. For example, the cost of a processor with 2 ns clock speed (*e.g.* Cray-3) is far more than 10 times the cost of a processor with 20 ns speed (*e.g.* Intel i860) given today's technology. Hence, it may be more cost effective to get higher performance by having a large number of relatively less utilized processors than to have a small number of processors whose speed is beyond the current technology threshold.

7 Concluding Remarks

Significant progress has been made in identifying and developing measures of scalability in the last few years. These measures can be used to analyze whether parallel processing can offer desired performance improvement for problems at hand. They can also help guide the design of large-scale parallel computers. It seems clear that no single scalability metric would be better than all others. Different measures will be useful under different contexts and further research is needed along several directions. Nevertheless, a number of interesting properties of several metrics and parallel systems have been identified. For example, a parallel system can be used to solve arbitrarily large problem instances in a fixed time if and only if its isoefficiency is linear. While solving a problem on a parallel computer, in order to make efficient use of the processors, it is necessary that the number of processors be governed by the isoefficiency function. If more processors are used, the problem might be solved faster, but the parallel system will not be utilized efficiently. For a wide class of parallel systems identified in [11], using more processors than determined by the isoefficiency function *does not* help in reducing the parallel time complexity of the algorithm.

The introduction of hardware cost factors (in addition to speedup and efficiency) in the scalability analysis is important so that the overall cost effectiveness can be determined. The current work in this direction is still very preliminary. Another problem that needs much attention is that of partitioning of processor resources among different problems. If two problems are to be solved on a multicomputer and one of them is poorly scalable while the other one has very good scalability characteristics on the given machine, then clearly, allocating more processors to the more scalable problem tends to optimize the efficiency, but not the overall execution time. The problem of partitioning becomes more complex as the number of different computations is increased. Some early work on this topic has been reported in [5, 31, 28, 27].

References

[1] S. Arvindam, Vipin Kumar, and V. Nageshwara Rao. Floorplan optimization on multiprocessors. In *Proceedings of the 1989 International Conference on Computer Design (ICCD-89)*, 1989. Also published as MCC Tech Report ACT-OODS-241-89.

[2] S. Arvindam, Vipin Kumar, and V. Nageshwara Rao. Efficient parallel algorithms for search problems: Applications in vlsi cad. In *Proceedings of the Frontiers 90 Conference on Massively Parallel Computation*, October 1990.

[3] S. Arvindam, Vipin Kumar, V. Nageshwara Rao, and Vineet Singh. Automatic test pattern generation on multiprocessors. *Parallel Computing*, 1991 (to appear). A short version appears in the proceedings of the International Conference on Knowledge-Based Systems, 1989, New Delhi, India.

[4] M.L. Barton and G.R. Withers. Computing performance as a function of the speed, quantity, andthe cost of processors. In *Proceedings Supercomputing '89*, pages 759 – 764, 1989.

[5] Krishna P. Belkhale and Prithviraj Banerjee. Approximate algorithms for the partitionable independent task scheduling problem. In *Proceedings of the 1990 International Conference on Parallel Processing*, pages I–72 to I–75, 1990.

[6] Chandran and Davis. An approach to parallel vision algorithms. In R. Porth, editor, *Parallel Processing*. SIAM, 1987.

[7] William Dally. Wire-efficienct vlsi multiprocessor communication network. In *Stanford Conference on Advanced Research in VLSI Networks*, pages 391–415, 1987.

[8] D.L. Eager, J. Zahorjan, and E.D. Lazowska. Speedup versus efficiency in parallel systems. *IEEE Transactions on Computers*, 38(3):408–423, 1989.

[9] Horace P. Flatt and Ken Kennedy. Performance of parallel processors. *Parallel Computing*, 12:1 – 20, 1989.

[10] Anshul Gupta and Vipin Kumar. On the scalability of Matrix Multiplication Algorithms on parallel computers. Technical report, Computer Science Department, University of Minnesota, Minneapolis, MN 55455, 1991.

[11] Anshul Gupta and Vipin Kumar. Analyzing performance of large scale parallel systems. Technical report, Computer Science Department, University of Minnesota, Minneapolis, MN - 55454, May 1991.

[12] Anshul Gupta and Vipin Kumar. On the scalability of FFT on parallel computers. In *Proceedings of the Frontiers 90 Conference on Massively Parallel Computation*, October 1990. An extended version of the paper is available as a technical report from the Department of Computer Science, and as TR 90-20 from Army High Performance Computing Research Center, University of Minnesota, Minneapolis, MN 55455.

[13] John L. Gustafson. Reevaluating Amdahl's Law. *Communications of the ACM*, 31(5):532–533, 1988.

[14] John L. Gustafson, Gary R. Montry, and Robert E. Benner. Development of parallel methods for a 1024-processor hypercube. *SIAM Journal on Scientific and Statistical Computing*, 9 No. 4:609–638, 1988.

[15] D. Hillis and G. L. Steele. Data parallel algorithms. *Communications of the ACM*, 29 (12):1170–1183, 1986.

[16] Jing-Fu Jenq and Sartaj Sahni. All pairs shortest paths on a hypercube multiprocessor. In *Proceedings of the 1987 International Conference on Parallel Processing*, pages 713 – 716, 1987.

[17] Laveen Kanal and Vipin Kumar (editors). *Search in Artificial Intelligence*. Springer-Verlag, New York, 1988.

[18] Alan H. Karp and Horace P. Flatt. Measuring parallel processor performance. *Communications of the ACM*, 33, No. 5:539 – 543, 1990.

[19] Clyde P. Kruskal, Larry Rudolph, and Marc Snir. A complexity theory of efficient parallel algorithms. Technical Report RC13572, IBM TJ-Watson Research Center, NY, 1988.

[20] Vipin Kumar and Anshul Gupta. Analyzing scalability of parallel algorithms and architectures. Technical report, TR-91-18, Computer Science Department, University of Minnesota, June 1991. A short version of the paper appears in the Proceedings of the 1991 International Conference on Supercomputing, Germany.

[21] Vipin Kumar and Anshul Gupta. Analyzing the scalability of parallel algorithms and architectures: A survey. In *Proceedings of the 1991 International Conference on Supercomputing*, June 1991.

[22] Vipin Kumar and V. N. Rao. Scalable parallel formulations of depth-first search. In Vipin Kumar, P. S. Gopalakrishnan, and Laveen Kanal, editors, *Parallel Algorithms for Machine Intelligence and Vision*. Springer-Verlag, New York, 1990.

[23] Vipin Kumar and V. Nageshwara Rao. Parallel depth-first search, part II: Analysis. *International Journal of Parallel Programming*, 16 (6):501–519, 1987.

[24] Vipin Kumar and V. Nageshwara Rao. Load balancing on the hypercube architecture. In *Proceedings of the 1989 Conference on Hypercubes, Concurrent Computers and Applications*, pages 603–608, 1989.

[25] Vipin Kumar and Vineet Singh. Scalability of Parallel Algorithms for the All-Pairs Shortest Path Problem. *Journal of Parallel and Distributed Processing (special issue on massively parallel computation)*, October 1991. A short version appears in the Proceedings of the International Conference on Parallel Processing, 1990.

[26] H.T. Kung. Memory requirements for balanced computer architectures. In *Proceedings of the 1986 IEEE Symposium on Computer Architecture*, pages 49 – 54, 1986.

[27] Eva Ma and Denis G. Shea. Downward scalability of parallel architectures. In *Proceedings of the Third International Conference on Super Computing*, pages 109 – 120, 1988.

[28] Lawrence W. Dowdy Michael R. Leuze and Kee Hyun Park. Multiprogramming a distributed-memory multiprocessor. *Concurrency: Practice and Experience*, 1(1):19 – 33, September 1989.

[29] V. Nageshwara Rao and Vipin Kumar. Parallel depth-first search, part I: Implementation. *International Journal of Parallel Programming*, 16 (6):479–499, 1987.

[30] David M. Nicol and Frank H. Willard. Problem size, parallel architecture, and optimal speedup. *Journal of Parallel and Distributed Computing*, 5:404 – 420, 1988.

[31] Kee Hyun Park and Lawrence W. Dowdy. Dynamic partitioning of multiprocessor systems. *International Journal of Parallel Processing*, 18, No. 2:91 – 120, 1989.

[32] Vineet Singh, Vipin Kumar, Gul Agha, and Chris Tomlinson. Scalability of parallel sorting on mesh multicomputers. In *Proceedings of the Fifth International Parallel Processing Symposium*, March 1991. Extended version available as a technical report (number TR 90-45) from the department of computer science, University of Minnesota, Minneapolis, MN 55455, and as TR ACT-SPA-298-90 from MCC, Austin, Texas.

[33] Xian-He Sun and L. M. Ni. Another view of parallel speedup. In *Proceedings Supercomputing '90*, pages 324–333, 1990.

[34] Zhimin Tang and Guo-Jie Li. Optimal granularity of grid iteration problems. In *Proceedings of 1990 International Conference on Parallel Processing*, pages I–111 – I–118, 1990.

[35] Fredric A. Van-Catledge. Towards a general model for evaluating the relative performance of computer systems. *The International Journal of Supercomputer Applications*, 3, No. 2:100 – 108, 1989.

[36] Patrick H. Worley. The effect of time constraints on scaled speedup. *SIAM Journal on Scientific and Statistical Computing*, 11 No. 5:838–858, 1990.

[37] Patrick H. Worley. Limits on parallelism in the numerical solution of linear pdes. *SIAM Journal on Scientific and Statistical Computing*, January 1991.

[38] Xiaofeng Zhou. Bridging the gap between amdahl's law and sandia laboratory's result. *Communications of the ACM*, 32 No. 8:1014–5, 1989.

[39] J.R. Zorbas, D.J. Reble, and R.E. VanKooten. Measuring the scalability of parallel computer systems. In *Proceedings of SuperComputing'89*, pages 832 – 841, 1989.

Type and Sharing Analysis for Concurrent Data Structures

Suresh Jagannathan
NEC Research Institute
Princeton, NJ 08540
suresh@research.nec.com

Abstract

Communication in asynchronous parallel programming languages often takes place through concurrent data structures (CDS). Such structures act as synchronization repositories for a collection of concurrently executing processes. Concurrent data structures, however, are rarely the focus of compile-time analysis.

In this paper, we present a novel compile-time analysis technique for implementing higher-order CDS-based languages. Our approach involves the construction of a type and sharing analysis framework for a kernel language that provides support for general first-class concurrent data structures; it is easily seen that the essence of many other CDS-based languages can be translated into this kernel.

The analysis we provide has two main purposes: (1) it ensures type correctness of explicitly parallel programs that define CDS objects, and (2) it answers questions regarding the implementation of these objects, and the communication properties of the processes which manipulate them.

Rather than restricting a type to be a set of values, we generalize the type of a concurrent data structure to be a set of *properties* that characterize its usage. The type of a CDS object contains information on the kinds of operations performed on it, the identity of the processes which manipulate it, and the constituent types of the elements it contains. Knowledge of such information can significantly improve the implementation of these structures as well as providing useful metrics for data placement, process scheduling, etc.

The type system is novel insofar as it operates in the presence of mutable, incrementally defined first-class CDS objects and non-flat heteregenous data structures. Because the work described addresses compile-time analysis of explicitly parallel programs within the context of a rigorous abstract interpretation framework (*i.e.*, type inference), we argue that it offers a promising approach to efficiently implement a variety of symbolic high-level parallel programming languages.

1 Introduction

Communication in asynchronous parallel programming languages often takes place through concurrent data structures (CDS). Such structures act as synchronization points for a collection of concurrently executing processes. Almost all explicitly parallel languages support some form of concurrent data structure – the stream object in Flat Concurrent Prolog[15, 16], objects created by the *future* construct in MultiLisp[10], the abstract store in Atomic Herbrand[14], the blackboard object in Shared Prolog[2], Concurrent Smalltalk's distributed objects[11], and C-Linda's flat tuple-space[6] are a few notable examples. Operationally, a concurrent data structure permits many consumers simultaneously to access its value; in some definitions, producers may also simultaneously augment or modify its contents. The semantics of these structures also specifies that consumers which attempt to access an as-of-yet undefined element are to block until a producer provides a value.

Concurrent data structures, however, are rarely the focus of compile-time analysis. Instead, many concurrent systems often rely on (a) runtime support or customized hardware to improve inefficiencies that may arise from the underlying (low-level) representation chosen for the abstract (high-level) structure (*e.g.,* [1, 9, 3, 8]), or (b) user-specified annotations to aid the compiler in generating optimal code (*e.g.,* [4, 7, 12]). Both approaches entail some significant cost: in the former case, the utility of concurrent data structures is constrained by the availability of specific hardware or operating-system support; in the latter case, programs must often be explicitly tailored to conform to a specific machine topology. In either case, abstraction and/or portability is compromised for efficiency's sake.

In this paper, we present a new compile-time analysis technique for implementing higher-order CDS-based languages. Our approach involves the construction of a type and sharing analysis framework for a kernel language that provides support for general first-class concurrent data structures; it is easily seen that the essence of many other CDS-based languages can be translated into this kernel. The type system is novel insofar as it operates in the presence of mutable, incrementally defined first-class CDS objects and non-flat heteregenous data structures. Because the work described addresses compile-time analysis of explicitly parallel programs within the context of a rigorous abstract interpretation framework (*i.e.,* type inference), we argue that it offers a promising approach to efficiently compile a variety of symbolic high-level parallel programming languages.

The analysis has two main purposes: (1) it ensures type correctness of explicitly parallel programs that define CDS objects, and (2) it answers questions regarding the implementation of these objects, and the communication properties of the processes which manipulate them.

Rather than restricting a type to be a set of values, we generalize the type of a concurrent data structure to be a set of *properties* that characterize its usage. The type of a CDS object reflects the operations performed on it, the identity of the processes which manipulate it, and the constituent types of the elements it contains. Knowledge of such information can significantly improve the implementation of these structures as well as providing useful metrics for data placement, process scheduling, etc. In addition, using type information to capture structural properties of CDS objects makes it possible to formally describe and implement optimizations on these structures without resorting to *ad hoc* methods.

2 Motivation

To motivate our discussion, we introduce a simple kernel language called $\mathcal{TS}^k$. For the purposes of the paper, the sequential core of $\mathcal{TS}^k$ consists of operators for defining function abstraction (λ), recursion (`letrec`), sequencing (`begin` and ;), and conditionals (`if`). This core forms a strict, higher-order functional language. In the full paper, we also extend our analysis to work in the presence of non-flat heterogeneous list structures *a la* Lisp.

Parallelism is expressed in $\mathcal{TS}^k$ via a collection of operators that build and access first-class *tuple-space* objects[1]. By way of summary, a tuple-space defines a content-addressable shared associative memory. The elements of a tuple-space are ordered sets of values or processes known as *tuples.* Fields in passive tuples, *i.e.,* tuples containing only values, can be retrieved using a simple pattern-matching procedure. Bindings established as a consequence of a pattern match are used to evaluate a subordinate expression. Failure to derive a suitable match causes the process that initiated the match operation to block. A blocked process may resume only after a matching tuple is deposited into the specified tuple-space. An active tuple, *i.e.*, a tuple containing concurrently executing processes, turns into a passive one when each process completes and returns a value. Active tuples are not involved in the matching procedure.

---

[1] Although we use tuple-spaces as a representative concurrent data structure in this paper, the ideas underlying our analysis can be readily applied to other CDS-based languages.

A first-class tuple-space is a tuple-space that can be (a) bound to variables, (b) passed as an argument to (or returned as a result from) other functions, or (c) embedded within data structures or tuples. Since tuple-spaces are first-class objects, they can be embedded within tuples; equality comparison on tuple-spaces is "eq"-based in the Lisp sense.

Figure 1 is a naive $\mathcal{TS}^k$ program fragment that defines a producer/consumer protocol via a bounded buffer in which tuple-spaces are used to represent the shared buffer object as well as the repository for shared state variables. The program fragment defines two procedures, `producer` and `consumer` that asynchronously generate and remove values from a shared buffer object defined in terms of a tuple-space; state variables used to determine the current head and tail of the buffer are stored in a separate tuple-space, `counter`; synchronization variables are kept in tuple-space `synch`, and processes are stored in tuple-space `process`.

Tuple-space operations (`get`, `rd`, `put` and `spawn`) all take the tuple-space they operate upon as an explicit argument. Operations that read (`rd`) or remove (`get`) tuples induce a pattern-match of their tuple argument against the tuples found in their argument tuple-space. Variables prefixed by "?" (called *formals*) are initially unbound and acquire a binding value as a consequence of the pattern-match. For example, the initial `get` operation found in `producer` binds `tl` to the second component of a tuple whose first field is `'tail`; if there exist several tuples with this structure, one is chosen arbitrarily. The binding established for `tl` is used in the evaluation of `get`'s body.

We note some interesting structural properties of this program fragment:

1. The first field in tuples found in `counter` are symbols drawn from the set, {`'tail`, `'head`}; the first field in tuples found in `synch` are drawn from the set, {`'consumer`, `'producer`}. Tuples involving `'tail` are modified only by `producer` processes, and tuples involving `'head` are modified only by `consumer` processes.
2. Only the second field in `counter` and `synch` is used in pattern matching operations. This field is constrained to be an integer in `counter` and a symbol in `synch`.
3. Instantiations of `consumer` modify `counter` tuples indexed by `'head`, but only read `counter` tuples indexed by `'tail`; similarly, `producer` instances manipulate `'tail` tuples, but only read `'head` ones.
4. The variable `v` used in `get` operations on `synch` in both `producer` and `consumer` is never referenced.
5. `Buffer` consists of two-tuples indexed by its integer-valued first field. Tuples are retrieved from `buffer` only via `get`; there are no `rd` operations performed on this tuple-space.
6. Write operations on `buffer` occur only in `producer`; remove operations occur only in `consumer`.
7. Both `producer` and `consumer` are instantiated as processes in tuple-space `process` and share access to `synch`, `counter`, and `buffer`. There are no read or remove operations on tuples stored in `process`.

In the absence of such information, a naive implementation would need to not only represent the various CDS objects defined in this fragment in terms of an associative lookup table (since tuples are accessed by content), but would also be unable to optimize the structure of this table based on the types of its constituent elements. Moreover, such an implementation can make no judgments on the scheduling of the processes that access these objects. In the presence of this information, however, we can construct the following representations for the tuple-spaces defined:

```
let synch   = (make-ts),
    counter = (make-ts),
    process = (make-ts),
    buffer  = (make-ts),

    producer = (λ ()
                  loop forever
                     (get counter ('tail,?tl)
                       (begin
                         (put buffer (tl,some value));
                         (get synch ('consumer,?v)
                              (put synch ('consumer,'proceed)));
                         let new_tl = (1+ (mod tl n))
                         in (begin
                               (put counter ('tail, new_tl));
                               (if (= new_tl (rd counter ('head,?hd) hd))
                                   (get synch ('producer,'proceed)
                                         (put synch ('producer,'wait))));
                               some value))))

    consumer = (λ ()
                  loop forever
                     (get counter ('head,?hd)
                       (begin
                         (if (= hd (rd counter ('tail,?tl) tl))
                             (get synch ('consumer, 'proceed)
                                  (put synch ('consumer,'wait)));
                         let result = (get buffer (hd,?elt) elt)
                         in (begin
                               (put counter ('head, (mod (1+ hd) *max-size*));
                               (get synch ('producer, ?v)
                                    (put synch ('producer,'proceed)));
                               result))))))
     in
      (begin
        (put counter ('tail,0));
        (put counter ('head,0));
        (put synch ('consumer,'wait));
        (put synch ('producer,'wait));
        (spawn process ((producer) (consumer))))
```

Figure 1: A producer/consumer program fragment using tuple-spaces as the communication medium. Let is used as syntactic sugar for application.

1. `Counter` is represented as two queues each with dedicated read and write locks. Operations on `'head` access one queue, and operations on `'tail` access the other. Producers can therefore increment the tail without needing to synchronize with consumers updating the head.
2. `Synch` is represented as a two vectors each of length two. Producers access one vector, and consumers access the other. The elements of these vectors are semaphores; `'proceed` and `'wait` are used to index into the appropriate vector. `Put` operations increment the appropriate semaphore while `get` operations decrement it. The operation

   `(get synch (`*producer/consumer*`, ?v)` *body*`)`

   is implemented by testing the value of both `'proceed` and `'wait` (in arbitrary order) decrementing whichever entry is non-zero, and blocking if both are zero.
3. `Buffer` is implemented as a vector with associated read and write locks. `Put` operations on the tuple-space abstraction are translated into write operations on the representation.
4. Instances of `producer` can always be scheduled before instances of `consumer` because `producer` augments `buffer` while `consumer` only accesses it; both procedures access and modify `synch` and `counter`.
5. The `process` tuple-space is not implemented in terms of a data structure since no operations on passive tuples are performed on it. Thus, `spawn` operations translate to simple process calls on the constitutent elements of its tuple argument.

Note that the utility of these analyses is not restricted to just tuple-space languages. Issues of data/process placement and eliminating unnecessary synchronization among processes is important in any parallel language that provides support for dynamic process creation (*e.g.*, [5, 10, 11]) and asynchronous communication. Since the patterns of usage of synchronization structures in these languages is also not explicitly declared, we anticipate analysis techniques of the kind described in this paper to be useful in these other contexts as well.

3 The Type System

In order to statically determine the structure of a tuple-space, we define a type system that associates every $\mathcal{TS}^k$ expression with a type; notably, the type of a tuple-space will reflect the structural characteristics of the tuples which inhabit it. Thus, to determine properties of tuple-spaces useful in building a customized representation, we examine their types.

The type language is shown in figure 2. ι ranges over primitive types, α ranges over tuple-space type variables (TSVars), and β ranges over type variables representing other types. Besides tuple-space and tuple types, there are two other type constructors in our type language: $\rightarrow$ builds function types, and $\times$ builds cross products.

The type language subsumes an attribute language used to characterize tuple-space objects. The base elements in the attribute language consist of the "null" attribute (written ϕ), primitive kinds (*e.g.*, READ, WRITE, etc.), pairs of the form: $< process - id, kind >$, triples of the form: $< process - id, integer, type >$, and four-tuples of the form: $< process - id, integer, expression, type >$. *Process-id* ranges over a set of unique identifiers denoting process labels.

Complex attributes are built from a sum constructor (+). Attributes capture specific properties of a tuple-space. Thus, $\mathcal{K}$ ranges over pairs consisting of kinds and the process labels of those processes that perform tuple-operations associated with this kind on the tuple-space; $\mathcal{F}$ ranges over structures containing the tuple indices of all formals (*i.e.*, "?" prefixed variables) occurring in `rd` and `get` operations, their types, and the process labels of those processes that perform `rd` and `get` operations on the tuple-space; $\mathcal{A}$ ranges over structures containing the non-formal

$$
\begin{array}{lcl}
\tau & ::= & \sigma \mid \iota \mid \tau \rightarrow \tau \mid \gamma \\
\sigma & ::= & \alpha \mid \beta \\
\gamma & ::= & (\delta, \mathcal{K}, \mathcal{F}, \mathcal{A}) \\
\delta & ::= & \tau \mid \delta \times \delta \\
\mathrm{S} & ::= & \phi \mid \textsc{read} \mid \textsc{write} \mid \textsc{remove} \mid \textsc{fork} \\
\mathrm{L} & ::= & \text{an infinite set of unique identifiers} \\
\mathcal{K} & ::= & \phi \mid (\mathrm{L}, \mathrm{S}) + \mathcal{K} \\
\mathcal{F} & ::= & \phi \mid (\mathrm{L}, \mathrm{N}, \tau) + \mathcal{F} \\
\mathcal{A} & ::= & \phi \mid (\mathrm{L}, \mathrm{N}, E, \tau) + \mathcal{A}
\end{array}
$$

Figure 2: The Type Language.

expressions, their locations, types, and the process labels of those processes that perform tuple operations containing non-formal expressions on this tuple-space.

For any tuple-space type γ and sum type, $\beta_1 + \beta_2 + \ldots\ \beta_n$ associated with attribute A found in γ, there exists a predicate $\downarrow_A$ such that $\gamma \downarrow_A \beta_i$ is true for $1 \leq i \leq n$ and false otherwise. We also adopt a canonical reduction that eliminates duplication of elements in a sum type: $\tau + \tau = \tau$.

4 A Type Algorithm

4.1 Dependency Propogation

In this section, we define a type assignment algorithm that computes a type for every $\mathcal{TS}^k$ expression. In particular, the algorithm addresses the issue of how to compute types for shared concurrent data structures that are subject to implicit side-effects.

To illustrate the problem, consider the following $\mathcal{TS}^k$ program fragment:

```
let f = (λ (g x)
           (begin
             (g x);
             x)),
    ts = (make-ts)
in
   (begin
     (put (f (λ (z) (put z ('foo)))
             ts)
          ('bar))
     ts)
```

(Here again, `let` syntax is used as sugar for application.)

The first argument to `put` in the `let`-body is an application of `f`. There is a side-effect induced by this application – namely, `f`'s functional argument deposits the tuple containing the single element `'foo` into its tuple-space argument. Because tuple-spaces are shared, and by the definition of `f`, the effect of the `put` operation on `z` must be reflected in `ts`. Thus, after evaluating the `let` expression, `ts` will contain two single-element tuples containing `'foo` and `'bar`.

To ensure that the type algorithm properly handles side-effects, and that it does not underspecify a tuple-space's attributes, we need to introduce an explicit notion of sharing. The type algorithm makes use of a relation called a *tuple dependency* for this purpose. Tuple dependencies capture the fact that type information inferred about one tuple-space object must get propagated to all other objects with whom it may be possibly aliased. Thus, in the above example, `ts` 's type is a tuple dependent of `z` 's.

4.2 Unification

Define a *substitution* to be a map from type variables to types. In the following, we sometimes apply substitutions to complex types or type environments; the meaning of such applications is taken to be the natural one – each type variable in the complex type or type environment is mapped to its type as defined by the substitution.

If S_1 and S_2 are substitutions, then $S_1 \circ S_2$ is the functional composition of S_1 and S_2. Extensions of substitutions are defined in the obvious way. Id denotes the identity substitution.

We define a variant of Robinson's unification procedure[13], $\mathcal{U}$, which given a pair of types τ_1 and τ_2, either fails or returns a type substitution such that:

1. If $\mathcal{U}(\tau_1, \tau_2)$ returns S, then $S\tau_1 = S\tau_2$.
2. If S unifies τ_1 and τ_2, and $\mathcal{U}(\tau_1, \tau_2)$ returns some substitution V, then there is another type substitution R such that $S = RV$. Moreover, V involves only type variables in τ_1 and τ_2.
3. If $\gamma_1 = (\delta_1, K_1, F_1, A_1)$ and $\gamma_2 = (\delta_2, K_2, F_2, A_2)$ then $\mathcal{U}(\gamma_1, \gamma_2)$ if $\mathcal{U}(\delta_1, \delta_2)$[2].

4.2.1 Tuple Dependencies

Define a *tuple dependency*, D, to be a map from TSvars to a set of TSvars:

$$D : TSvars \rightarrow \mathcal{P}(TSvars)$$

The null type dependency D_ϕ maps its argument to the empty set.

We define the join of two tuple dependencies as follows: $(D_1 \circ_d D_2)(\alpha) = D_1(\alpha) \cup D_2(\alpha)$

We also define a procedure $\mathcal{R}$ that computes new tuple dependencies:

$$\mathcal{R}(\tau_1, \tau_2) = \begin{cases} D_\phi[\tau_1 \mapsto \{\tau_2\}, \tau_2 \mapsto \{\tau_1\}] & \text{if } \tau_1 \text{ and } \tau_2 \text{ are TSvars} \\ D_\phi & \text{otherwise} \end{cases}$$

4.3 A Type-Assignment Procedure

We are now in a position to present a type-assignment procedure for $\mathcal{TS}^k$. The procedure takes as its arguments a type environment ρ, a process id ψ, a tuple dependency D, and an expression e. It returns as its result a type substitution, a new tuple dependency, and a type τ for e.

Algorithm $\mathcal{TC}$

$\mathcal{TC}(\rho, \psi, D, e) = (S, D', \tau)$

1. if e is x and $\rho(x) = \tau'$ then $S = Id$, $D' = D$ and $\tau = \tau'$.

---

[2] This rule in effect states that the success of unifying two tuple-space types does not depend on their structural attributes, but only on the types of the tuples they contain.

```
2. if $e = (\lambda\ (x)\ e_1)$, then

       let $\sigma$ be a new type variable,
           $\mathcal{TC}(\rho[x \mapsto \sigma], \psi, D, e_1) = (S_1, D_1, \tau_1)$
       in  $S = S_1$, $D' = D_1$, and $\tau = S_1(\sigma) \to \tau_1$

3. if $e = (\texttt{letrec}\ x\ =\ e_1\ \texttt{in}\ e_2)$, then

       let $\sigma$ be a new type variable,
           $\mathcal{TC}(\rho[x \mapsto \sigma], \psi, D, e_1) = (S_1, D_1, \tau_1)$
           $\mathcal{TC}(S_1[\sigma \mapsto \tau_1](\rho), D_1, e_2) = (S_2, D_2, \tau_2)$
       in  $S = S_2$, $D' = D_2$, and $\tau = \tau_2$

4. if $e = (e_1\ e_2)$, then

       let $\mathcal{TC}(\rho, \psi, D, e_1) = (S_1, D_1, \tau_1 \to \tau_2)$,
           $\mathcal{TC}(S_1(\rho), \psi, D_1, e_2) = (S_2, D_2, \tau_3)$,
           $\mathcal{U}(S_2(\tau_1 \to \tau_2), S_2(\tau_3) \to \sigma) = S_3$    (where $\sigma$ is fresh)
           $\mathcal{R}(\alpha, \tau_3) = D_3$    (where $\alpha$ is fresh)
           $S_f = S_1 \circ S_2 \circ S_3$
           $S' = \begin{cases} S_f[\alpha \to S_f(\tau_1)] & \text{if } \tau_1 \text{ is a TSvar} \\ S_f & \text{otherwise} \end{cases}$
       in  $S = S'$, $D' = D_2 \circ_d D_3$ and $\tau = S_3(\sigma)$

5. if $e = (\texttt{make} - \texttt{ts})$, then

       let $\alpha$ be a new TSvar
           $\sigma$ be a new type variable
           $D(\alpha) = \{\alpha\}$
       in  $S = Id[\alpha \mapsto (\sigma, \phi, \phi, \phi)]$, $D' = D$, and $\tau = \alpha$

6. if $e = (\texttt{put}\ e_1\ (e_2))$, then

       let $\mathcal{TC}(\rho, \psi, D, e_1) = (S_1, D_1, \alpha)$
           $\mathcal{TC}(S_1(\rho), \psi, D_1, e_2) = (S_2, D_2, \tau_1)$
           $S_f = S_1 \circ S_2$
           $S_f(\alpha) = (\tau_1, K, F, A)$
           $S'(\sigma) = \begin{cases} S_f(\sigma) & \text{if } S_2(\sigma) \notin D_2(\alpha) \\ (\tau_1, K + (\psi, \text{WRITE}), F, A + (\psi, 1, e_2, \tau_1)) & \text{otherwise} \end{cases}$
       in  $S = S'$, $D' = D_2$, $\tau = S'(\alpha)$

7. if $e = (\texttt{spawn}\ e_1\ (e_2))$, then

       let $\mathcal{TC}(\rho, \psi, D, e_1) = (S_1, D_1, \alpha)$
           $\psi_1$ be a fresh process label
           $\mathcal{TC}(S_1(\rho), \psi_1, D_1, e_2) = (S_2, D_2, \tau_1)$
           $S_f = S_1 \circ S_2$
           $S_f(\alpha) = (\tau_1, K, F, A)$
           $S'(\sigma) = \begin{cases} S_f(\sigma) & \text{if } S_2(\sigma) \notin D_2(\alpha) \\ (\tau_1, K + (\psi, \text{EVAL}), F, A + (\psi, 1, e_2, \tau_1)) & \text{otherwise} \end{cases}$
       in  $S = S'$, $D' = D_2$, $\tau = S'(\alpha)$
```

8. if $e = (\texttt{rd}\ e_1\ (e_2)\ e_b)$, then

$$\begin{aligned}
\text{let } & \mathcal{TC}(\rho, \psi, D, e_1) = (S_1, D_1, \alpha) \\
& \mathcal{TC}(S_1(\rho), \psi, D_1, e_2) = (S_2, D_2, \tau_1) \\
& S_f = S_1 \circ S_2 \\
& S_f(\alpha) = (\tau_1, K, F, A) \\
& S_b(\sigma) = \begin{cases} S_f(\sigma) & \text{if } S_2(\sigma) \notin D_2(\alpha) \\ (\tau_1, K + (\psi, \text{READ}), F, A + (\psi, 1, e_2, \tau_1)) & \text{otherwise} \end{cases} \\
\text{in } & \mathcal{TC}(S_b(\rho), \psi, D_2, e_b)
\end{aligned}$$

9. if $e = (\texttt{rd}\ e_1\ (?x)\ e_b)$, then

$$\begin{aligned}
\text{let } & \mathcal{TC}(\rho, \psi, D, e_1) = (S_1, D_1, \alpha) \\
& S_1(\alpha) = (\tau_1, K, F, A) \\
& S_f(\sigma) = \begin{cases} S_1(\sigma) & \text{if } S_1(\sigma) \notin D_1(\alpha) \\ (\tau_1, K + (\psi, \text{READ}), F + (\psi, 1, \tau), A + (\psi, 1, e_2, \tau_1)) & \text{otherwise} \end{cases} \\
\text{in } & \mathcal{TC}(S_f(\rho[x \mapsto \tau]), \psi, D_2, e_b)
\end{aligned}$$

10. otherwise return fail.

In typing an application, a new instantiation of the function's domain type is constructed; this type is passed as an argument along with the actual's type to $\mathcal{R}$. The result is a new tuple dependency if these types are tuple-space type variables. The reason for defining a new instantiation of the domain type is to ensure that different calls to the same function do not share tuple dependencies with one another.

`Put` and `rd` operations use the tuple dependency set associated with their tuple-space type to propogate new tuple information to other tuple-space types. Recall that the type variables found in a tuple dependency set of a tuple-space T correspond to tuple-spaces that share the same tuple structure as T. `Spawn` operations define new process labels that are used in typing the component elements of their tuple argument.

5 Flow Analysis

The same analysis used to detect structural properties of a tuple-space can be generalized to infer interesting flow patterns of $\mathcal{TS}^k$ programs. Space limitation prevent us from describing the approach in detail, but we provide a summary here. We can define a lattice of flow attributes whose primitive components include ϕ, READ, WRITE, REMOVE, READ/WRITE, WRITE/READ, WRITE/REMOVE, etc. These attributes are associated with expressions. Non-tuple-space expressions have flow attribute ϕ; tuple-space operations augment the flow attribute of their tuple-space argument. A tuple-space with attribute WRITE/READ indicates that regardless of the flow of control taken in the program, all writes on the tuple-space are guaranteed to take place before any reads. This analysis must necessarily be conservative, and in the case where two processes simulatenously operate upon the same tuple-space object in different ways, the flow analysis may yield no useful information. On the other hand, empirical evidence indicates that for well-structured programs with little non-determinism, one can assign meaningful flow attributes to tuple-spaces. The flow analysis interpretation is structurally similar to the type analysis except for the constants in the flow language and the subsumption relationship that exists between them. For example, a WRITE/READ attribute is weaker than a WRITE attribute; the interpretation applied assigns the strongest attribute to tuple-space expressions that preserve this relation.

References

[1] A. Agrawal, B-H. Lim, D. Kranz, and J. Kubiatowicz. APRIL: A Processor Architecture for Multi-Processing. In *Proceedings of the* 17^{th} *Annual International Symposium on Computer Architecture*, pages 104–115, May 1990.

[2] V. Ambriola, P Ciancarini, and M. Danelutto. Design and Distributed Implementation of the Parallel Logic Language Shared Prolog. In *Second ACM Symposium on Principles and Practice of Parallel Programming*, pages 40–49, March 1990.

[3] Arvind and Rishiyur Nikhil. Executing a Program on the MIT Tagged-Token Dataflow Architecture. In *Proceedings of the PARLE Conference*, 1987. Published as Springer-Verlag LNCS Volume 259.

[4] Rajive Bagrodia and Sharad Mathur. Efficient Implementation of High-Level Parallel Programs. In *Proceedings of the Fourth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, pages 142–151, 1991.

[5] H.E. Bal and A.S. Tanenbaum. Distributed Programs with Shared Data. In *Proceedings of the IEEE International Conference on Computer Languages*, pages 82–91, 1988.

[6] Nick Carriero and David Gelernter. Linda in Context. *Communications of the ACM*, 32(4):444 – 458, April 1989.

[7] Marina Chen, Young-Il Choo, and Jingke Li. Compiling Parallel Programs by Optimizing Performance. *Journal of Supercomputing*, 2:171–207, 198.

[8] William Dally *et. al.* The J-Machine: A Fine-Grain Concurrent Computer. In *Proceedings of the 1989 IFIPS Conference*, 1989.

[9] R.H. Halstead and T. Fujita. MASA: A MultiThreaded Processor Architecture for Parallel Symbolic Computing. In *Proceedings of the* 15^{th} *Annual International Symposium on Computer Architecture*, pages 443–451, June 1988.

[10] Robert Halstead. Multilisp: A Language for Concurrent Symbolic Computation. *Transactions on Programming Languages and Systems*, 7(4):501–538, October 1985.

[11] Waldemar Horwat, Andrew Chien, and William Dally. Experience with CST: Programming and Implementation. In *ACM SIGPLAN '89 Conference on Programming Language Design and Implementation*, pages 101–109, June 1989.

[12] Charles Koelbel and Piyush Mehrotra. Supporting Shared Data Structures on Distributed Memory Machines. In *Second ACM Symposium on Principles and Practice of Parallel Programming*, pages 177–187, March 1990.

[13] John A. Robinson. A Machine-Oriented Logic Based on the Resolution Principle. *Journal of the ACM*, 12:23 -44, 1965.

[14] Vijay Saraswat. *Concurrent Constraint Programming Languages.* MIT Press, 1989. Also published as a CMU Ph.D Thesis, January 1989.

[15] Ehud Shapiro. Concurrent Prolog: A Progress Report. *IEEE Computer*, 19(8):44–60, August 1986.

[16] Ehud Shapiro, editor. *Concurrent Prolog : Collected Papers.* MIT Press, 1987. Volumes 1 and 2.

An Object-Oriented Concurrent Reflective Architecture for Distributed Computing Environments

Yuuji Ichisugi, Satoshi Matsuoka, Takuo Watanabe, and Aki Yonezawa
Department of Information Science, Faculty of Science,
University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo Japan 113
{*ichisugi,matsu,takuo,yonezawa*}*@is.s.u-tokyo.ac.jp*

1 Introduction

Reflection is the process of reasoning about and acting upon the system itself[6, 4, 10]. In a conventional computation system, computation is performed on data that represent (or model) entities which are external to the system. In contrast, a reflective computation system must contain some structure that represents (or models) the structural and computational aspects, namely the metalevel information, of itself. Such a representation, called *self-representation*, must be *dynamically* accessible and modifiable within the system itself.

Previously, programming language functionalities such as debugging facilities, interfaces to external world, and exception handling are added to languages in an ad hoc and inflexible manner. In contrast, languages based on reflective execution models are powerful in providing linguistic mechanisms to use the above functionalities in a uniform and flexible manner [4, 3].

The capability of dynamically manipulating the self-representation of the structure and computation process allows the user to improve the execution efficiency of the base-level computation, the computation that is the primary goal of the application. The dynamic optimization in the framework of reflective computation is especially effective in a distributed concurrent computation system because (1) the reflective structure allows the programmer to dynamically control the computational aspects such as scheduling, communication, and load-balancing, that do not arise in sequential computing [1], and (2) both the reflective computation and the baselevel computation can be performed in parallel[10, 5, 7].

We realized the above significance of reflective computation in a distributed concurrent computation system [7, 10] and proposed two basic frameworks for reflective computation: the Individual-Based Architecture (IBA) [7] and the Group-Wide Architecture (GWA) [8] [2]. Furthermore, we recently combined these two approaches as our new framework, the Hybrid Group Architecture (HGA) [5]. It realizes inter-object cooperation and optimization through dynamic control of resource shared by objects. We designed and implemented three languages ABCL/R, ACT/R, and ABCL/R2 that correspond to the

---

[1] One of the interesting applications is to dynamically change the scheduling policy of object execution [1] in distributed simulation based on the Time Warp Scheme [2].

[2] The original idea of GWA was mentioned in the last sections of [7] and [10].

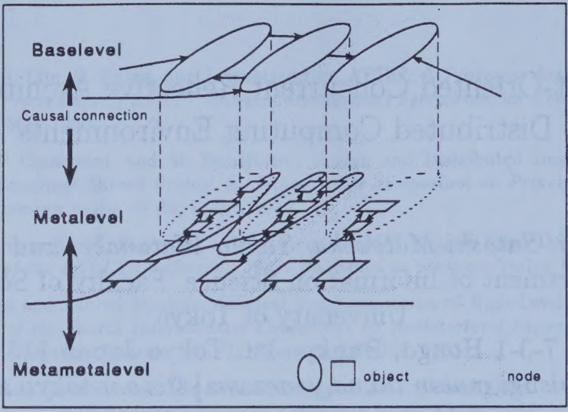


Figure 1: The baselevel and the metalevel

three frameworks. We demonstrated the effectiveness of the reflective computation in concurrent system through several experiments.

The main motivation of the current research is to develop a practically useful concurrent programming system that is easy to customize and become adaptable. We propose our object-oriented concurrent reflective language RbCl (Reflection-based Concurrent language). The implementation of the RbCl system should work efficiently for practical use and satisfy the following requirements:

1. The system should be adaptable to various environments that differ in the architecture of parallel computers, and in the characteristics of applications. The system should also be able to evolve according to dynamic changes of execution environments.

2. The system should be used as a platform for various experiments on parallel languages to test linguistic mechanisms and implementation schemes.

3. The programmer should be able to access external environments (e.g., window systems, database systems, etc.) in a uniform manner.

The requirements 1 and 2 are realized by the reflective computation facilities RbCl provides. To make our language easy to program and practically useful, one of our goals of this research is to provide a *disciplined* scheme for the usage of reflective facilities that is analogous to the metaobject protocol (MOP) of CLOS [3].

The *metalevel architecture* of the RbCl system is similar to that of the HGA. Each unit of the metalevel representation models (or represents) a group of baselevel objects, and such a group corresponds to the group of objects running on a single processor node. The RbCl metalevel architecture models the actual hardware configuration of a distributed computation system. Basically, the *metalevel architecture* consists of *metalevel objects* that form a metacircular definition of the execution mechanism of baselevel objects, just as ABCL/R2. Namely, the execution mechanism of a group of objects is represented in terms of a (unit) group of metalevel objects. Reflective computation in RbCl is realized by the interaction between the baselevel objects and the metalevel objects as in [7, 8, 5]. (Figure 1). Our *metaobject protocol* (RbCl-MOP) to be designed provides a set of rules to customize and modify/extend metalevel objects in a disciplined manner.

One of the interesting characteristics of the RbCl system is the linguistic symbiosis with other object-oriented languages (e.g., C++). This notion is different from "external language interface" of previous languages. The linguistic symbiosis in RbCl is based on the metalevel architecture, and both RbCl objects and C++ objects symbiotically reside in a single system interdependently. This symbiosis enables us to construct an efficient implementation of the RbCl system itself and also enables us to use external systems (e.g., window systems and database system, etc.) in a uniform manner.

C++ objects and RbCl objects may coexist in the RbCl system and they can communicate bidirectionally with each other. An RbCl object can treat a C++ object just as an RbCl object, and conversely, a C++ object can treat an RbCl object just as a C++ object. Thus in the communication among objects of the two languages, each language specific communication protocol is used individually. An RbCl object communicates to a C++ object via the RbCl message passing protocol, and a C++ object communicates to an RbCl object by the C++ virtual function invocations.

This functionality allows the programmer to replace a (default) RbCl metalevel object with a metalevel object coded in C++ with the same behavior. The current implementation of the RbCl system uses this technique to improve the execution efficiency of the system. In fact, it is such an implementation that enables the co-existence of RbCl objects and C++ objects. This interdependency between the two languages at the metalevel characterizes the linguistic symbiosis of RbCl.

The next section describes the language RbCl and its metalevel architecture in more details. Section 3 discusses the relationship between RbCl and C++. Section 4 illustrates the current implementation scheme of the RbCl system. Finally, we summarize our work and present the future direction in Section 5.

2 The Language RbCl and its Metalevel Architecture

The language RbCl is similar to our object-oriented concurrent programming language ABCL/1 [9] to a certain extent. Each object is the unit of concurrency and the operations performed within an object is executed based on a sequential imperative computation model. Objects are dynamically created and they interact each other only by sending messages (we do not assume shared variables). RbCl provides three modes of message passing: 1) *past (asynchronous) mode* message passing does not wait for a reply message, 2) *now mode* message passing blocks the execution of the sender object until it receives the reply, and 3) by sending in *future mode*, the sender can get the contents of the reply without being blocked in the future access. RbCl lacks some linguistic facilities of ABCL/1 such as the `wait-for` statement and the *express mode* message passing.

As stated in Section 1, the baselevel is the world where objects perform usual object-level computation and the metalevel is the world which constitutes the language execution system including the runtime support facilities. The objects that form the metalevel are called the *metalevel objects*. In the RbCl model, metaobjects realizes the computation of the objects at the baselevel by communicating with each other at the metalevel(Figure 1). Thus, the metalevel is the implementation of the baselevel, which makes the *causal connection* [4] maintained in a natural way. The computation in the metalevel can also be controlled by the computation in the meta-meta level. Therefore it is possible in our model to have an infinite tower of metalevels.[3]

---

[3] Actually, the meta-meta level does not exist in the initial setting. See Section 4.

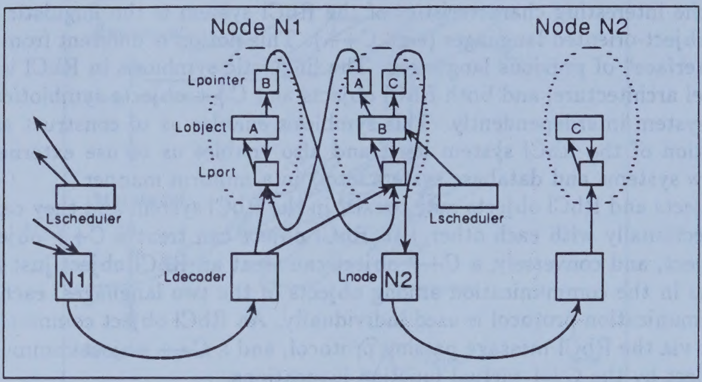


Figure 2: The metalevel of the RbCl system

The RbCl system is designed to function in a distributed environment and the assumed hardware architecture is the network of nodes each being possibly a composite of tightly coupled CPUs. In programming the baselevel, the programmer need not be aware of the distributed nature of the architecture. However, the design of the metalevel takes into account of the node boundaries. In the metalevel representation corresponding to a node, all the entities that constitute the RbCl system (e.g., message queues of objects, pointers to objects, the scheduler, etc.) are all defined as RbCl metalevel objects and the programmer can manipulate these metalevel objects in a *disciplined manner* (by following the rules defined in the RbCl-MOP), and can easily modify/extend various schemes for distributed or parallel execution such as message passing protocols and scheduling policy. In designing the metalevel architecture in terms of RbCl metalevel objects, we tried to partition it into two parts, one that depends on the hardware architecture and the other that does not, so that the RbCl system becomes portable to other system architectures and adaptable to the combination with other parallel languages.

2.1 The structure of the metalevel

Figure 2 illustrates several important metalevel objects:

- Lobject
  For each baselevel object, there is an *Lobject* that contains all the information required to implement the corresponding baselevel object. For example, as its instance variables, it contains the metalevel objects which represent the method code and evaluation environment for variable binding.

- Lport, Lpointer
  For each baselevel object, there is an *Lport* which is an input port of messages delivered to the baselevel object and it contains a message queue to store delivered messages. An *Lpointer* is an abstraction of a global pointer. It contains the network-wide global representation of the address of an Lport and realizes network transparent communication. Referencing to the port of an object is accomplished by an indirect reference through the Lpointer which represents the address of the referenced object.

- Lnode, Ldecoder

  An *Lnode* sends *encoded* messages to the *Ldecoder* that resides in another node in the network. The Ldecoder interprets the messages it receives, and corresponding to the contents of the messages, it may create objects in the node it resides, or may send messages to the objects in the same node. In the default configuration of the metalevel, referencing and message passing beyond the node boundaries are performed only through Lnodes and Ldecoders.

- Lscheduler

  An *Lscheduler* sends a **run** message to an executable instance of an Lobject in now mode. Every Lobject is coded to give the thread of control back to the scheduler after the execution of its method code for a while. The scheduler successively sends **run** messages to schedulable objects in the node and performs non-preemptive scheduling, thus achieving pseudo parallelism.

2.2 Reflection

The programmer can manipulate the metalevel objects by following the RbCl-MOP and change the behavior of the RbCl system in order to cope with different characteristics of applications and dynamic changes in the computing environment. For example, the programmer can change the scheduling policy of the Lobjects from the default round-robbin scheduling to some priority scheduling by replacing the default scheduling queue of the Lobjects with a priority queue. By using reflective facilities, the programmer can add/modify other functionalities such as migration, persistent objects, distributed garbage collection, and debugging facilities.

The realization of the reflection in the RbCl system is made possible by the following architectural properties:

1. Objects at the metalevel can be referenced (as first class objects) by objects at the baselevel,
2. Objects at the baselevel can dynamically create objects at the metalevel,
3. Message passing between the baselevel and metalevel is possible.

In creating a baselevel object, if non-default Lobject and Lport are specified (instead of the default ones) for the baselevel objects, it is possible to change the semantics of message passing involving the baselevel object. (E.g., message passing can be changed from ordinary one to a priorized one.)

Because RbCl provides the same linguistic functionality at both the baselevel and metalevel, objects at both levels can send *inter-level* messages asynchronously. Consequently, the computations performed at the baselevel and metalevel can execute in parallel. An object can send messages to metalevel objects even in other nodes, network-transparently. Also objects can change the behavior of the metalevel by manipulating objects at the meta-meta level.

2.3 Inter-node communication

Communication beyond the node boundaries is realized by the functions of Lnodes. To make our system flexible, the communication protocol should be extensible and modifiable. For example, when the programmer wishes to use a new distributed garbage collection algorithm to the application, he/she needs more sophisticated information to be passed through the communication protocol, whose extensibility is required. Moreover, when the programmer wants to reduce the message traffic by efficiently compressing information, he/she needs to encode the information in an application-specific way, which requires the modifiability of communication protocol.

The flexibility (extensibility and modifiability) in inter-node communication is obtained by the use of a protocol description language called NaCl (Node action Control language). This language is simple and the system can interpret it efficiently. NaCl commands are interpreted by sending the message to the corresponding metalevel objects. The programmer can dynamically add a new command to an Ldecoder in another node by sending an RbCl program code to the Ldecoder, creating a new metalevel object. Consequently, NaCl has the same expressivity as RbCl, in spite of its simplicity,.

The programmer can use NaCl not only for the communication protocol description but also for the description of the formats for persistent objects, RbCl library code, etc.

3 Linguistic Symbiosis of RbCl Objects and C++ Objects

Most reflective languages designed to date are homogeneous languages in that they do not have well-defined communication protocols with other programming languages. However, having external interfaces to other languages is necessary for practical use.

RbCl provides the functionality to hide the semantical gap between RbCl and C++ and enables communication between an RbCl object and a C++ object. Namely, RbCl objects and C++ objects can symbiotically co-exist in the RbCl system. This enables the following three facilities:

- Linkage between the RbCl system and C++ libraries, which allows access to object-oriented databases, window systems, and operating systems,
- Tuning-up of the component objects of the RbCl system which are critical for high execution efficiency, and
- Efficient implementations of the metalevel (Subsection 4.1)

The C++ implementation of an RbCl object must meet the following requirements:

1. A C++ class must be defined to be a descendent of the class L. To access a C++ object, the RbCl system needs to invoke the methods provided by the class L .
2. All the methods which the RbCl may invoke must be defined as virtual functions.
3. All the arguments of the methods which the RbCl may invoke must have the type either basic type (e.g., integer, character, etc.) or the pointer type referencing an instance of the class L.
4. There are some methods that every class definition must contain. An example is class_name.

These requirements do not weaken the expressiveness of C++. The RbCl preprocessor can analyzes the C++ class definitions and automatically generates the requisite methods mentioned in the above 4th requirement.

C++ objects satisfying those requirements can be viewed as ordinary RbCl objects from the other RbCl objects because C++ objects behave just the same as ordinary RbCl objects: the RbCl system can send messages to them and treat them as first class objects. Conversely, C++ objects can view an RbCl object as an instance of L (or a descendent of L) and may send messages by virtual function invocations.

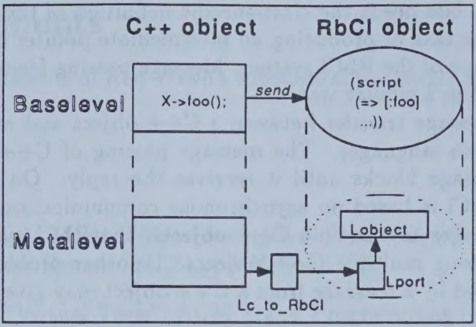


Figure 3: Message passing from a C++ object to an RbCl object

4 Implementation of the RbCl System

4.1 Metalevel and Reflection

For high execution efficiency of the metalevel, we implement every metalevel object to be a C++ object which fulfills the requirements listed in Section 3. The default metalevel objects accept only messages sent in now mode (message passing in now mode corresponds to a virtual function call of C++) and they can be implemented by C++. The default inter-node communication protocol supports only byte stream communication, which can be implemented using the inter-node communication primitives the underlying operation system provides.

In this design of the metalevel, the initial setting of the system does not require meta-meta level objects which are needed for the dynamic evolution of the system (e.g., changing the form of parallel execution or the network transparent communication protocols). Meta-meta level objects are created in a lazy manner when an RbCl object is created at the metalevel by reflective operations.

The functionality needed for reflection such as creation of an RbCl object at the metalevel and message passing from baselevel objects to metalevel objects can be naturally realized by the functionality of message transmission between RbCl objects and C++ objects.

4.2 Message Transmission between RbCl Objects and C++ Objects

Figure 3 illustrates how a message sent by a C++ object is delivered to an RbCl object. Viewing from the baselevel, it looks that a C++ object directly references an RbCl object. However, at the metalevel, the RbCl object is indirectly referenced through a metalevel object called *intermediate pointer*. The method foo defined for the intermediate pointer class looks like:

```
Lc_to_RbCl::foo(){
   // Send the RbCl message [:foo] to the port.
   port->send(list(TAG_FOO));
}
```

This method translates the C++ representation of a message and its arguments into the RbCl representation. Because instances of different C++ classes receive different set of messages, every class needs the class-specific definition of the intermediate pointer class. This burdensome task of producing an intermediate pointer class is automatically done by the preprocessor of the RbCl system. Message passing from an RbCl object to a C++ object is realized in a similar way.

One problem in message transfer between a C++ object and an RbCl object is the semantic gap of the two languages. The message passing of C++ is synchronous and the sender of the message blocks until it receives the reply. On the other hand, the message passing of RbCl is based on asynchronous communication. Because one RbCl object may send messages to multiple C++ objects, the RbCl system has to emulate pseudo parallelism among multiple C++ objects. Another problem is that the RbCl object which is activated by a message from a C++ object may give the thread of control to another object.

We use a light-weight thread library to realize pseudo parallelism among multiple C++ objects. All the RbCl objects in the same node works inside the *scheduler thread* and inter-RbCl message passing does not cause the thread switching. When an RbCl object sends a message to a C++ object, the system creates a new thread whose stack is used to process the received message. When a C++ object sends a message to an RbCl object, the C++ object blocks and returns the thread of control back to the scheduler thread. When the RbCl object returns the reply, the thread of the C++ object is reactivated. Communication between C++ objects is implemented by ordinary C++ method invocations efficiently.

5 Summary

We have proposed an object-oriented concurrent *reflective* language RbCl and its implementation design that are efficient enough for practical use in distributed environments.

The programmer has flexible control over the system to (dynamically) change the behavior of the system following the rule the RbCl-MOP imposes. The metalevel of RbCl is composed of metalevel objects which implement useful facilities for distributed languages. Especially, the programmer can easily extend and modify functionalities for distributed and/or parallel execution systems such as message passing protocols and object scheduling policy. Consequently, the system adapts itself to various parallel hardware architectures and various application characteristics, and the system can dynamically evolve according to execution environments. The RbCl system can also used as a platform to test the design and implementation of parallel languages.

One of the novel features of RbCl is its provision of the functionalities for linguistic symbiosis of different languages. For example, RbCl objects and C++ objects can coexist in the same system, and transfer messages to each other beyond the semantic gap of the two languages. The capability of calling C++ virtual functions from an RbCl object improves the execution efficiency of the application and allows the programmer to link the application with existing C libraries and also to call the low-level functionalities that the operating system provides. Moreover, by coding the metalevel of the RbCl in C++, the resulting RbCl system becomes very fast.

The current implementation of the RbCl is begin done on the message passing parallel computer iPSC/2, and SparcStation 2 workstations connected to network (Ethernet). A prototype metalevel implementation has been completed and the system runs in a distributed environment. We would like to complete the full-spec implementation of the system and also search for an elegant definition of the RbCl-MOP.

Acknowledgments

The illuminating comments of Ken Wakita substantially strengthened the work presented in this paper.

References

[1] Burdorf, C. and J. Marti, "Non-Preemptive Time Warp Scheduling Algorithms", Operating Systems Review, vol.24,pp7-18,Apr.1990.

[2] Jefferson, D.R., "Virtual Time", ACM Trans. Programming Languages and Systems, vol.7,no.3,pp.404-425, 1985.

[3] Kiczales, G., des Rivieres, J. & Bobrow, D. G., The Art of Metaobject Protocol, MIT Press, 1991.

[4] Maes, P., "Concepts and Experiments in Computational Reflection", In Proc. ACM OOPSLA '87, pp. 147-155, 1987.

[5] Matsuoka, S., Watanabe, T. & Yonezawa, A., "Hybrid Group Reflective Architecture for Object-Oriented Concurrent Reflective Programming", in Proc. ECOOP '91, pp. 231–250, Lecture Notes in Computer Science, 512, Springer, 1991.

[6] Smith, B. C., "Reflection and Semantics in Lisp", In Conference Record of ACM POPL '84, pp. 23-35, 1984.

[7] Watanabe, T. & Yonezawa, A., "Reflection in an Object-Oriented Concurrent Language", In Proc. ACM OOPSLA '88, pp. 306-315, 1988, (revised version in [9]).

[8] Watanabe, T. & Yonezawa, A., "An Actor-Based Metalevel Architecture for Group-Wide Reflection", In Proc. REX School/Workshop on Foundations of Object-Oriented Languages (REX/FOOL), Noordwijkerhout, the Netherlands, May 1990, Also in Lecture Notes in Computer Science, 489, Springer, 1991.

[9] Yonezawa, A.(Ed.), ABCL: An Object-Oriented Concurrent System, MIT Press, 1990.

[10] Yonezawa, A. & Watanabe, T., "An Introduction to Object-Based Reflective Concurrent Computations", In Proc. of 1989 Workshop on Object-Based Concurrent Programming, ACM SIGPLAN Notices, 24(4), pp. 50-54, 1989.

Hyperprograms: Concurrent Automata as Information Structures*

P. David Stotts
Department of Computer and
Information Sciences
University of Florida
Gainesville, FL 32611

Richard Furuta
Department of Computer Science and
Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742

Abstract

Based on Petri nets, the Trellis model of hypertext captures not only the logical relationships among hypertext information elements, but also allows flexible specification of dynamic document properties. Multiple concurrent elements and paths, as well as multiple readers, can be expressed, synchronized, and otherwise managed directly by the parallel automaton that makes up a Trellis document.

A client-server software architecture is used to implement systems based on the Trellis model. The separation of form and function into visual clients and formal-automaton servers makes it especially easy for multiple users to interact in one Trellis information structure. In addition, different automata can be used for different classes of structures, without changing the interface clients.

In this paper we present several variants of Petri Nets that we are using as formal automata for Trellis documents, and discuss our adaptation of Clarke's *model checking* algorithms to the verification of browsing properties expressed by these automata.

Key words: hypertext, Petri nets, browsing semantics, concurrency, synchronization, security, temporal logic, verification, access control, model checking, human-computer interaction, parallel finite automata.

1 The problem and the approach

Among others, Halasz has noted the need for structural search and query mechanisms for increasing the utility of hypertext nets [6]. He identifies two major subtasks: "... to design a query language geared toward describing hypermedia network structures," and "... implementation of a search engine capable of satisfying the queries expressible in the new language." This report describes an approach to both of these subtasks.

---

[*] This work is based upon work supported by the National Science Foundation under grant numbers IRI-9007746 and IRI-9015439.

We concentrate on a mechanism for answering queries concerning dynamic properties a document has, that is, what sequences of links a reader may be allowed to follow during browsing, but the basic technique, that of *model checking* (borrowed from concurrent system verification), can be easily adapted to locating readers within the structure under examination.

1.1 Summary of the Trellis model

The Trellis model has been formally defined elsewhere [9, 10], though we include a brief informal description here to aid the current discussion. Abstractly, a Trellis hypertext (document) is an annotated timed Petri net. The information associated with the places is *document content*. The annotations on the transitions are adaptation agents (small chunks of Lisp code). The net itself is the *static document structure*. The state space of the net defines the *dynamic document structure*.

Browsing the document is analogous to executing the Petri net, except that the normally nondeterministic transition firings are directed with selections made by the readers of a document. During browsing, when a token enters an empty place, the content of that place (if any) is presented for consumption. Any enabled transitions are shown to the readers as selectable links (buttons). If a reader clicks on a button, the associated transition is fired and tokens are moved around. The Lisp agent associated with the transition (if any) is also executed.[1]

A Trellis system is a particular realization of the Trellis model with a client-server software architecture. The Petri net model is implemented as an engine that has no visible interface but will respond to RCP calls for its services. Clients are written to present a visual interface, but they have no inherent behavior; rather, they provide user access to the automaton that defines the behavior. Each document is an abstract parallel automaton. Several clients can check in to a document, allowing multiple users to interact in the information structure. The particular semantics of multi-user interaction is defined solely by the server automaton. For example, if the server implements classical Petri nets, then each reader will see the same state of the document, and actions by one reader will cause all other readers to have their view altered as well. If, however, the server is a colored Petri net, then the document can differentiate one reader from another by color, and can provide more sophisticated interactions.

We have implemented αTrellis with a classical (timed) Petri net engine and have done most of the experimentation with this system. Currently we are building χTrellis and σTrellis based on a colored (timed) Petri net engine, for use in collaboration support systems for astronomy image browsing and software process modeling respectively.

---

[1] The Lisp agents are used to record information about a readers choices so that changes to document parameters, such as transition timings, can be made to adapt the document behavior to a reader's preferences [11]. We will not concern ourselves further with them in this paper.

2 HTL*: Hypertext Temporal Logic

Treating a hyperdocument as an automaton, parallel or otherwise, is a novel approach. The traditional abstraction is a static directed graph, with browsing behavior for a document defined by the code making up each specific hypertext system. When a hyperdocument is viewed as an automaton instead of a graph, the potential state sequences the automaton can produce during "execution" defines the dynamic behavior of the document. This behavior is a property of each document, not of the system implementation, and this behavior has a succinct and formal definition in the firing rule of the automaton. We refer to such Trellis information structures, combining the nature of automata with hypertextual interaction, as *hyperprograms*.

Petri nets were chosen as the automata for hyperprograms for several reasons: first, parallelism is inherent and succinctly expressible; next, potentially infinite states allow a richer range of expressible functions than with finite state automata, yet without "tapes" or other memory abstractions that do not fit nicely into a hypertext framework; finally, several variants of Petri nets exist (timed, colored, etc.) and have nicely developed analytical techniques.

Since the browsing behavior is of paramount importance in hyperdocuments, it is natural to wis to allow authors to check their documents for the presence or absence of important behavior. Since we treat documents at programs, in essence, we cann adopt verification methods from concurrent program research, techniques that would not be obvious to proponents of the "static graph" hyperdocument abstraction. One such technique that we have successfully employed in Trellis is to express these properties in a branching temporal logic and then check the validity of the formulae on the Petri net.

In a branching temporal logic one introduces special logic symbols which allow formulation of assertions involving relative ordering as well as quantification over paths in the tree-like model of sequences. Various authors have proposed distinct syntaxes and semantics for branching time logics [7, 1], which differ in expressive power. However, CTL*, developed by Emerson and Halpern [5], is one of the most general languages for branching time logics, since it properly contains most of the other branching time logic languages. Moreover, CTL* finds a very natural interpretation in the context of the Trellis model of a hypertext.

We have defined a temporal logic language called HTL* specifically for hypertext authors to express browsing properties. HTL* has basically the operators of CTL* with some backwards path quantifiers similar to those of POTL [8]. The details of the HTL* syntax are beyond the scope of this short paper.

2.1 Verifying validity for HTL* formulae

We are not directly interested in a logical inference system for HTL*. That is to say, given a temporal formula, we are not in interested in proving that this formula is valid using axioms and inference rules. Our main goal, at least for the time being, is to present HTL* as a powerful and expressive temporal logic language especially suited for *automated* verification of browsing properties in Trellis hyperprograms. What we require in this context is a technique to determine if a given hypertext satisfies certain properties specified in HTL*. This kind of problem has already been extensively studied in other contexts [3], most notably in the domain of protocols specified by collections of finite state machines. These solutions are collectively referred to as *model checking*.

As a first cut at checking the validity of HTL* formulae on Trellis hyperprograms, we have taken Clarke's model checking algorithms for CTL (an efficiently checkable subset of CTL*) and have adapted our documents to their format. We have defined a language HTL that is a proper subset of HTL*; it is analogous to CTL as compared to CTL*. HTL permits only branching-time operators; each path quantifier must be immediately followed by exactly one of the operators *nexttime* ($\circ$) or $\mathcal{U}$. HTL could be consider the restriction of HTL* (which trivially subsumes pure linear and branching temporal logic) to its branching time dimension. Because of the backward path operators, HTL turns out to be *syntactically* equivalent to POTL rather than to CTL. POTL, proposed by Pinter and Wolper, is described in [8].

The language HTL is important in that it is easy to map its operators into CTL operators, to take advantage of the work that has been done for checking efficiently the validity of CTL formulae in a given model[3, 4, 2].

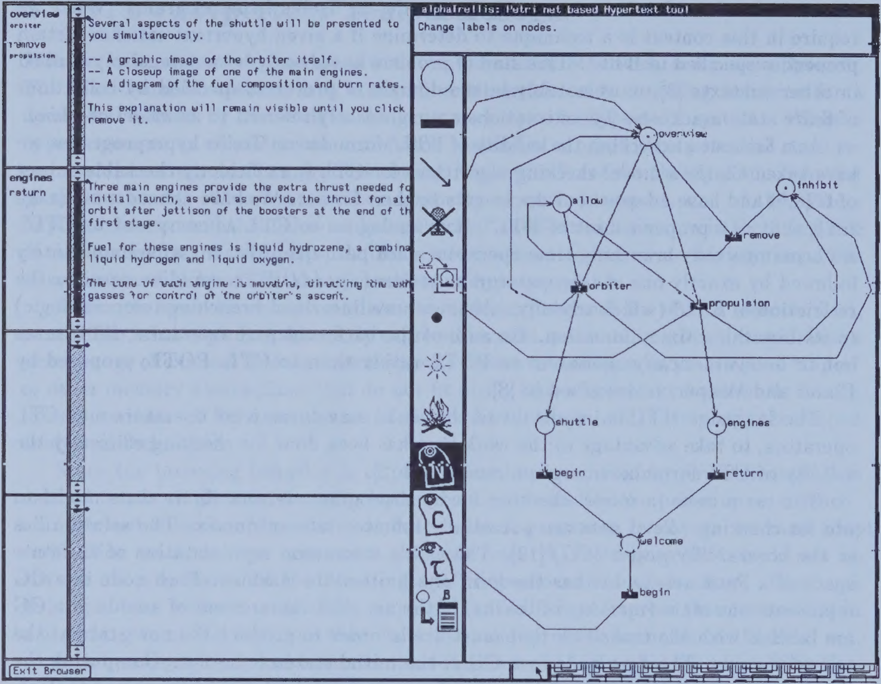
The main issue in model checking for hyperprograms is what finite state model to use for checking. Petri nets are potentially infinite state automata. The solution lies in the *coverability graph (CG)* [12]. The CG is a common representation of the state space of a Petri net, and it has the form of a finite state machine. Each node in a CG represents one state (marking of tokens) of the net, and the arcs out of a node in a CG are labeled with the transition that must fire in order to produce the new state at the end of the arc. The first node in a CG is the initial state of the net. Computing the CG for a net requires space and time that is exponentially related to the size of the net, but it is a terminating procedure for all nets. Since a CG is a finite representation of a possibly infinite state space, clearly some information is lost in constructing it. We discuss this problem in section 3.

2.2 Examples of interaction verification

In this section we show how the model checker developed at Carnegie Mellon by Clarke can be used to check temporal properties in the context of the Trellis model of hypertext. We will check HTL formulae (except for the backward path quantifiers) by expressing them in the corresponding CTL notation.

A simple Trellis document and sample queries

The Trellis document shown in figure 1 is a small net that expresses the browsing behavior found in some hypertext systems, namely that when a link is followed out of a node, the source content stays visible and the target content is added to the screen. The source must later be explicitly disposed of by clicking a "remove" button. The Petri


overview
orbiter
remove
propulsion
engines
return
Exit Browser
alphaTrellis: Petri net based Hypertext tool
overview
inhibit
allow
remove
orbiter
propulsion
shuttle
engines
begin
return
welcome
begin
N
C
τ
B

Several aspects of the shuttle will be presented to you simultaneously.

-- A graphic image of the orbiter itself.
-- A closeup image of one of the main engines.
-- A diagram of the fuel composition and burn rates.

This explanation will remain visible until you click remove it.

Three main engines provide the extra thrust needed fo initial launch, as well as provide the thrust for att orbit after jettison of the boosters at the end of th first stage.

Fuel for these engines is liquid hydrozene, a combina liquid hydrogen and liquid oxygen.

The cone of each engine is movable, directing the exh gasses for control of the orbiter's ascent.

Change labels on nodes.

Figure 1: Small αTrellis document.

net shown gives rise to a CG of 11 states.. From this, we generate a finite state machine in the format needed by Clarke's CTL model checker. This model can then be queried for the desired browsing properties with the model checker. We illustrate several here, using the syntax for Clarke's CTL model checker, and showing its output.

- Is there some browsing path such that at some point both the "orbiter" and "propulsion" buttons are selectable on one screen?

```
|= EF(B_orbiter & B_propulsion).
The formula is TRUE.
```

- Does there exist a browsing path such that at some point both the "shuttle" text and the "engines" text are concurrently visible?

```
|= EF(C_shuttle & C_engines).
The formula is FALSE.
```

- Is there some browsing path that reaches a point at which no buttons are selectable?

  ```
  |= EF( ~B_begin & ~B_remove & ~B_propulsion & ~B_orbiter
          & ~B_start & ~B_return ).
  The formula is FALSE.
  ```

- Is there some browsing path that will eventually simultaneously show the "overview" panel and the "engines" panel?

  ```
  |= EF(C_overview & C_engines).
  The formula is TRUE.
  ```

- Is it possible during browsing to see both the "welcome" and "engines" panels on the same screen?

  ```
  |= EF(C_welcome & C_engines).
  The formula is FALSE.
  ```

- Can both the "allow" access control and the "inhibit" access control ever be in force at the same time?

  ```
  |= EF(C_inhibit & C_allow).
  The formula is FALSE.
  ```

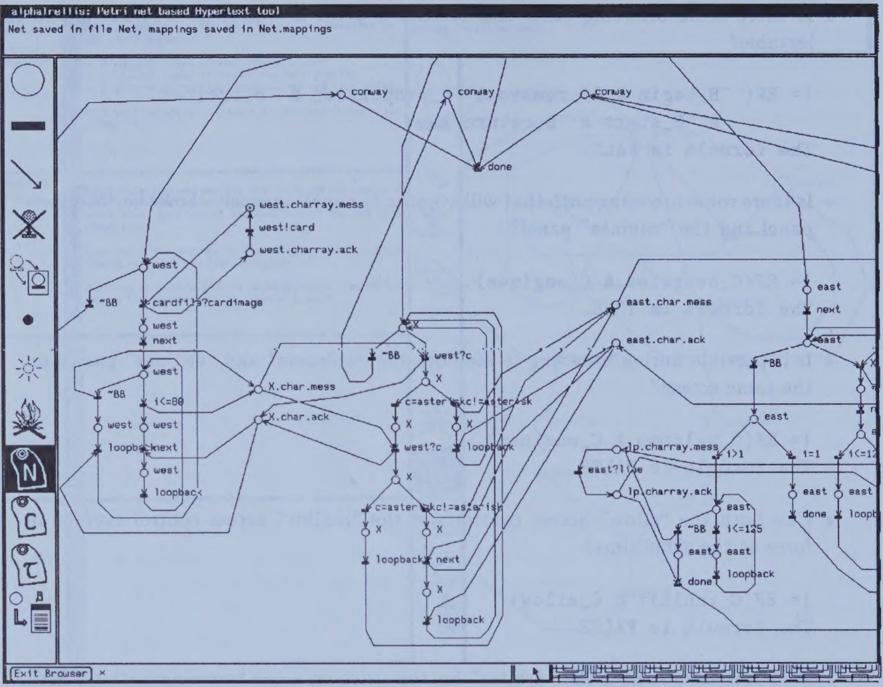
- Is it possible to select the "orbiter" button twice on some browsing path without selecting the "remove" button in between?

  ```
  |= EF(B_orbiter & AX(A[B_remove U B_orbiter])).
  The formula is FALSE.
  ```

This last query is a bit more complex. The informal expression of it does not parallel closely the actual operators employed to check the property.

A larger Trellis document

The model checker has also been tested on the larger Trellis document shown in Figure 2. The state machine derived from this net contains nearly a thousand states. The performance of the verification software on formulae like those discussed for the smaller example was on the order of a few seconds to nearly a half-minute, running on single processor of an Encore Multimax. We suspect that authors of interactive documents will find such timing not at all unreasonable for establishing the presence or absence of critical browsing properties, and we hope that future, less primitive systems, might exhibit improved performance as well.



alphaTrellis: Petri net based Hypertext tool

Net saved in file Net, mappings saved in Net.mappings

conway
conway
conway
done
west.charray.mess
west!card
west.charray.ack
west
~BB
cardfile?cardimage
west
next
west
X.char.mess
X.char.ack
~BB
i<=80
west
west
loopback
next
west
loopback
~BB
west?c
X
c=aster
kc!=aster
ask
X
X
west?c
loopback
X
c=aster
kc!=asterisk
X
X
loopback
next
X
loopback
east
east.char.mess
next
east.char.ack
east
~BB
east
lp.charray.mess
east?line
i>1
i=1
i<=12
lp.charray.ack
east
east
east
done
loopb
~BB
i<=125
east
east
done
loopback
Exit Browser

Figure 2: CSP program as browsable αTrellis document

3 Parallel Finite Automata

One of the problems with model checking for Petri nets is that a CG may have nodes that contain ω, that is, that represent an infinite number of actual net states. Since some information is discarded in constructing this ω graph from the original net, correct inference cannot always be done and the results of model checking may be inconclusive. For example (in a net of 3 places), if a query asked "is place 2 always marked immediately after place 1 is marked on any path" and a state sequence is found in the coverability graph that is "(2 1 4) → (2 ω 4)" then the query cannot be answered accurately. The ω represents an arbitrary number of tokens, which means it also represents zero tokens. Hence, the second state shown indicates that there *may* exist a state following the one with place 1 marked in which place 2 is *not* marked. There is no way from the CG alone to tell if that state is actually reachable or not.

One solution to this problem is to use a finite state model for hyperdocuments in the first place. Since the overt concurrency in Petri notation is a useful authoring device for information structures, we have created a new Petri-net variant we call Parallel Finite Automata (PFA) to provide a finite space while retaining a syntax similar to classical

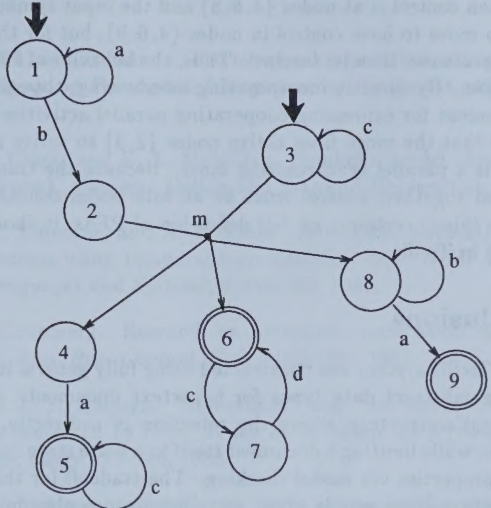


Figure 3: Another PFA example

Petri nets.

Informally, a PFA is simply a classical Petri net with a normalizing execution rule. A PFA starts with a token distribution that has *no more* that one token per place, and after a transition fires (using classical Petri net criteria) the token count in each place is normalized. Since no place will ever contain more than one token, the maximum number of states in a PFA is the number of distinct subsets of its places.

To illustrate the representation and execution of a PFA, consider the diagram in Figure 3. This PFA recognizes the language

$$(a^*b \parallel c^*)\ m\ (ac^* \parallel (cd)^* \parallel b^*a)$$

where the || operator specifies an interleaving of the languages that are its operands. It has two start nodes (1 and 3) and three final nodes (5, 6 and 9). An execution sequence for this PFA accepting the word "acbmaba$" is as follows:

Active Nodes	Input Symbol	Action
{1,3}	a	control from 1 to 1
{1,3}	c	control from 3 to 3
{1,3}	b	control from 1 to 2
{2,3}	m	control from {2,3} to {4,6,8}
{4,6,8}	a	control from 4 to 5
{5,6,8}	b	control from 8 to 8
{5,6,8}	a	control from 8 to 9
{5,6,9}	$	accept

Note that when control is at nodes $\{4, 6, 8\}$ and the input symbol is "a" the automaton can choose to move to have control in nodes $\{4, 6, 9\}$, but for the word in this example no accepting state can then be reached. Thus, the behavior of PFAs is somewhat similar to that of NFAs. By directly incorporating interleaving, though, the notation of PFAs is more convenient for expressing cooperating parallel activities.

Note also that the move from active nodes $\{2, 3\}$ to active nodes $\{4, 6, 8\}$ on input symbol "m" is a parallel synchronizing move. Because the transitions leaving nodes 2 and 3 are tied together, control must be at *both* nodes concurrently for the move to occur. While this is certainly no full definition of PFAs, it should give an idea of how they are used in Trellis.

4 Conclusions

The original Trellis system was constructed using fully general implementations of Petri net engines as abstract data types for hypertext documents. Using a PFA, though, for a document's structure allows the retention of a directly parallel hypertext link representation while limiting a document itself to a finite state space for exact provability of browsing properties via model checking. The tradeoff for this gain in verification is that Some expressive power is given up, since we are abandoning the more powerful class of Petri net for essentially finite state machines, We are still trying to classify the differences as far as authors are concerned between the two automata classes.

The implementation of a provable Trellis system based on PFAs was a straightforward modification of the original software system. Since the PFA form is basically the same as a Petri net, no modifications were required to document representations or to the manner in which information associated with net components is handled and presented. The DFA required to perform model checking is then directly available from the existing Trellis analysis facilities that compute coverability graphs for the nets comprising documents. In effect, the only change required to create the new system was a modification to the net execution rule, making it perform token normalization on each new state. In a system on the order of 20,000 lines of code, this change affected about 40 lines divided over two modules (the net simulation module and the state space analysis module). To use DFAs directly would have entailed a change to the fundamental data structures underlying the Trellis system, and would have therefore affected almost all modules.

We conclude by noting that, even though PFAs and DFAs are equivalent in language theoretic terms, the inherently parallel state representation a PFA makes this automaton class useful as a modeling tool. The fact that the class DFA is a direct *structural* subset of the class PFA makes parallel finite automata a preferred replacement for traditional the state machine representation in many applications (those involving any concurrency).

The client-server architecture of Trellis systems makes the adoption of new document engines, like PFAs, especially clean. Work is currently under way investigating the utility of other Petri net variants as well for hyperprograms.

References

[1] M. Ben-Ari, A. Pnueli, and Z. Manna. The temporal logic of branching time. In *In Proc. eighth ACM Symp. on Principles of Programming Languages*, pages 164–176, 1981.

[2] J. R. Bursh, E. M. Clark, and K. L. McMillan. Symbolic model checking: 10 to the 20 states and beyond. Carnegie Mellon and Stanford Universities, 1989.

[3] E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems*, 8:244–263, 1986.

[4] E. M. Clarke and Grumberg. Research on automatic verification of finite-state concurrent systems. *Ann. Rev. Comput. Sci.*, 2:269–290, 1987.

[5] E. A. Emerson and J. Y. Halpern. "sometimes" and "not never" revisited: on branching vs. linear time. In *In Proc. Tenth ACM Symp. on Principles of Programming Languages*, pages 127–140, 1983.

[6] Frank G. Halasz. Reflections on NoteCards: Seven issues for the next generation of hypermedia systems. *Communications of the ACM*, 31(7):836–852, July 1988.

[7] L. Lamport. "sometime" is sometimes "not never": on the temporal logic of programs. In *In Proc. seventh ACM Symp. on Programming Languages*, pages 174–185, 1980.

[8] Pinter and Wolper. A temporal logic for reasoning about partially ordered computations. In *In Proc. of the third ACM Symp. on Principles of Distributed Computing*, 1984.

[9] P. David Stotts and Richard Furuta. Petri-net-based hypertext: Document structure with browsing semantics. *ACM Transactions on Information Systems*, 7(1):3–29, January 1989.

[10] P. David Stotts and Richard Furuta. Temporal hyperprogramming. *Journal of Visual Languages and Computing*, 1(3):237–253, 1990.

[11] P. David Stotts and Richard Furuta. Dynamic adaptation of hypertext structure. In *Proceedings of Hypertext 91*, December 1991. To appear.

[12] Reisig Wolfgang. *Petri Nets an Introduction.* Springer-Verlag, 1985.

Programming with Composition[1]

Stephen Taylor
Department of Computer Science
California Institute of Technology

Abstract

The Scalable Concurrent Programming Project at Caltech is experimenting with the concept of programming by *composition*[1]. The goal of this research is to provide a formal and practical foundation for nondeterministic concurrent programming. The central idea is to focus on the methods by which programs are glued together or *composed.* Conventional sequential programming is fundamentally the use of two kinds of glue: sequential composition and choice (if-then-else) composition, variables provide a mechanism for communication between program units.

This talk discusses methods for composing non-deterministic concurrent programs. The ideas of program composition are described via examples, representative performance figures for applications in fluid dynamics are shown. The basic glues are defined on program units and the method by which these units are built is relatively unimportant. Thus, existing codes and libraries, written in Fortran, C, C++, Ada, etc., can be reused within new concurrent programs. An interesting aspect of this approach is that it is straightforward for programmers to define their own glues via semantic preserving source-to-source transformations. These correspond to abstractions that can be reused for a variety of applications within a specific domain, such as fluid dynamics, weather simulation or DNA sequencing.

The ideas have been implemented and are publically available in an experimental concurrent programming system. This system allows portable concurrent programs to be designed that operate on a broad range of multicomputers, shared memory machines and workstation networks. These include HP, Next and Sun workstations, Sequent Symmetry, Intel iPSC2 and Delta, and Amatek s2010. The system includes an optimizing compiler, performance analysis tools, program tracing tools, symbolic debugger, program mapping tools, and source-to-source transformation tools.

---

[1] This research is sponsored by the Defence Advanced Research Projects Agency, DARPA Order Number 8176, and monitored by the Office of Navel Research under contract number N00014-91-J-1986.

References

[1] [CT] Chandy, K.M. and Taylor S., "An Introduction to Parallel Programming", Jones and Bartlett, Boston, MA 02116, 1991.

*Algorithms for Resource Allocation in Multiple Service, Multiple Resource Communication Networks**

Scott Jordan
Department of Electrical Engineering
Northwestern University, Evanston, IL 60208-3118
scott@eecs.nwu.edu

Abstract

The merging of telephone and computer networks is introducing multiple resources into networks, and information is becoming increasingly distributed across the network. Related services are being integrated onto a single network rather than being offered on separate uncoordinated networks. In this paper, we focus upon communication networks that integrate multiple services using multiple resources. Of particular consideration is the numerical computation of quantities of interest, such as throughputs, blocking probabilities, average revenue, sensitivities, optimal and near-optimal control policies, near-optimal resource acquisition, and optimal prices.

Previous research efforts have used a discrete distribution. Since the state tracks the number of each service in the system, the number of states is on the order of the *product* of the capacities of each service in the system. Sums which run through each state in the space, therefore, can take a prohibitive computation time. In this paper, we consider approximating the discrete distribution by a continuous distribution, and base our algorithms upon this continuous distribution to evade these problems.

I. Introduction

In this paper, we focus upon communication networks that integrate multiple services using multiple resources. Of particular consideration is the numerical computation of quantities of interest, such as throughputs, blocking probabilities, average revenue, sensitivities, optimal and near-optimal control policies, near-optimal resource acquisition, and optimal prices.

This work is motivated by several trends in networks. The merging of telephone and computer networks is introducing multiple resources into networks, and information is becoming increasingly distributed across the network. Related services are being integrated onto a single network rather than being offered on separate uncoordinated networks.

These trends are made possible by the availability of fiber and of inexpensive electronic storage, and by the introduction of greater intelligence into the signaling system. Furthermore, these trends are made profitable by the proliferation of desktop computers and the increased demand for better information transfer.

Proposals for implementing services in these ***multiple service, multiple resource (MSMR)*** networks abound. A few examples of these **services** might be electronic/voice mail, mixed media telephone calls, video conferencing, distributed databases, hypertext systems, electronic catalogues, electronic yellow pages, and collaborative editors.

Our premise is that each *service* relies upon a number of underlying *resources* in the network. Examples of these **resources** might be communication links, databases, switches, storage devices, special purpose hardware and software. Although the precise meaning of "service" and "resource" and the relationship between them is a topic for future research, we assume in this paper that we have identified each service and the set of resources on which it depends.

Integrated services will share resources both for functionality and to decrease cost. Since these resources are limited, there will be interaction among the services. What types of interaction might we see? If you are the manager of a multiple service, multiple resource system, what requests for service do you accept? Based on what? If you base these decisions on maximizing revenue, what prices do

---

* This work was supported by the National Science Foundation under grant NCR-9109223, and by Ameritech under an Ameritech Research Fellowship.

you charge? And what resources should you acquire? The purpose of this research effort is to address such *resource allocation problems.*

In [Jord91], we investigated the nature of this interaction, and in [JordPR1], we investigated management of a MSMR system. In [JordPr2], we introduced a continuous version of the model used in the previous two papers. In this part, we address algorithmic considerations.

Several papers have addressed theoretical aspects of the problem. MSMR systems are considered in [Kell86/88] [Burm84]. Related but simpler *multiple service, single resource* (MSSR) systems are considered in [Aein78] [Kauf81] [Virt88] [RossPr] [NainPr] [Ross89a/b].

Numerical considerations have been directly principally at calculating blocking probabilities, from which quantities of interest in an *uncontrolled* system can be derived. The main problem is in calculating the normalization constant of the discrete distribution. Significant contributions and good overviews of the current status of this research effort can be found in [Whit85] [Kell86] [Tsan90].

Some effort has also applied to numerical computation of optimal or near-optimal control policies, from which quantities of interest in a *controlled* system can be derived. Finding control policies relies on first calculating blocking probabilities and then improving them, and is thus more computationally intensive than analysis of an uncontrolled system. Zachary and Key have investigated using policy improvement to iteratively find good policies in [Zach88] and [KeyPr].

Problems remain with all of these schemes in the analysis of large systems. Since the state tracks the number of each service in the system, the number of states is on the order of the *product* of the capacities of each service in the system. Sums which run through each state in the space, therefore, can take a prohibitive computation time. In this paper, we consider approximating the discrete distribution by a continuous distribution, and base our algorithms upon this continuous distribution to evade these problems.

In [Jord91], we investigated the MSMR model for the simplest discipline: a request is granted if the necessary resources are available; otherwise it is rejected. In [JordPR1], we considered strategies that may not accept all requests when the necessary resources are available, but may instead hold out for high paying requests. In [JordPr2], we introduced a continuous version of the discrete model originally presented in [Jord91] and the optimal coordinate convex control policy for this continuous model. We review these results in section II. Section III discusses algorithms based on the continuous model. In section IV, we show how these algorithms can calculate quantities of interest in a *uncontrolled* system. In sections V and VI, we show how these algorithms can calculate quantities of interest in a *controlled* system.

II. Background

A. The Discrete Model

Consider the following model for resource allocation in *uncontrolled* multiple service multiple resource (MSMR) communication networks:

Model:

Consider a system that offers n types of services. Each service requires a set of resources (dependent upon the service type) to process. If these resources are available then the system manager accepts a service request, and then processing starts immediately; if the necessary resources are unavailable then the request is lost to the system. (In later sections, we will allow the system manager to deny a service request even if the necessary resources are available.)

Service requests arrive as independent Poisson processes. Each request occupies each resource that it needs for the same amount of time, and releases these resources simultaneously upon service completion. This amount of time is exponentially distributed, and independent of other service times.

We model this system as a Markov chain. Adopt the following notation:

$\lambda \equiv (\lambda_1, \cdots, \lambda_n)$, the rates of incoming service requests.

$\mu \equiv (\mu_1, \cdots, \mu_n)$, the rates of service.

$\rho \equiv (\rho_1, \cdots, \rho_n)$, the loads, given by $\rho_i = \lambda_i / \mu_i$.

$L \equiv (L_1, \cdots, L_n)$, the rates of *accepted* service requests (throughput).

$x \equiv (x_1, \cdots, x_n)$, the state of the system, where

$x_i \equiv$ number of type i requests being processed.

$Z \equiv \{ x \mid x$ is feasible, i.e. x can be simultaneously processed with available resources $\}$

$F_i \equiv \{ x \mid x \in Z$ but $(x_1, \cdots, x_i+1, \cdots, x_n) \notin Z \}$, the full set w.r.t. service type i.

$E_i \equiv \{ x \mid x \in Z$ but $(x_1, \cdots, x_i-1, \cdots, x_n) \notin Z \}$, the empty set w.r.t. service type i.

$\pi(x)$, the steady state probabilities.

Our assumptions regarding the arrival and departure processes gives us a Markov chain on state space Z with transition rates:

$$r_{xy} = \begin{cases} \lambda_i, \text{ if } x \notin F_i \text{ and } y = (x_1, \cdots, x_i+1, \cdots, x_n) \\ x_i \mu_i, \text{ if } x \notin E_i \text{ and } y = (x_1, \cdots, x_i-1, \cdots, x_n) \\ 0, \text{ else} \end{cases}$$

Assume that service completion is never blocked. This implies that the state space Z is *coordinate convex*, i.e. if $x \in Z$ & $x_i \geq 1$, then $(x_1, \cdots, x_i-1, \cdots, x_n) \in Z$.

The Markov chain is time reversible with well known product form stationary distribution [Jord91]:

$$\pi(x) = \pi(0) \prod_{i=1}^{n} \frac{\rho_i^{x_i}}{x_i!} \qquad \pi(0) = \frac{1}{\sum_{x \in Z} \prod_{i=1}^{n} \frac{\rho_i^{x_i}}{x_i!}} \tag{1}$$

B. The Optimization Problem

We now assume that the system manager can choose to deny a service request even if the corresponding resources are available. Why might she do this? Since resources are limited, accepting a request of one type may preclude the possibility of accepting a request of another type in the near future. We restrict ourselves to control policies that are coordinate convex (c.c.). In this class, admission decisions depend on ***the state the system would enter if the request is granted***. This is equivalent to ***restricting the state to some subset of the original space***. Since service completion can not be realistically blocked, this subset, like the original state space, must be coordinate convex. This class has the nice property that the model of the controlled system is the restriction of the uncontrolled time reversible Markov chain to a subset, and is thus itself time reversible, maintaining its nice product form stationary distribution.

Adopt the following notation:

$r_i \equiv$ revenue generated by servicing request type i, per unit of time.

$r(x) \equiv$ rate of revenue generated while in state x, namely $\sum_i r_i x_i$

$R \equiv E\, r(X) = r(EX)$

$=$ the average revenue per time unit generated by the system.

Our goal is to maximize expected revenue. Our objective is thus to choose the c.c. subset $Z^* \subseteq Z$, that maximizes $E[\, r(X) \mid X \in Z^* \,]$. To talk about sets of states it is helpful to introduce some constructs that relate sets to each other:

Define the supporting set of V to be the minimum set required to make V c.c., namely:

$$ss(V) \equiv \{\, x \notin V \mid 0 \le x_i \le v_i \;\; \forall\, i \;\; \text{for some } v \in V \,\}$$

Define a projection function:

$$x{\uparrow}I \equiv (y_j), \;\; y_j = \begin{cases} x_j \,, & j \notin I \\ 0 \,, & j \in I \end{cases} \qquad \text{for } I \subset N.$$

Let N be the set of all service types {1, ... , n}. And define a cross section:

$$C_Z(x{\uparrow}I) \;=\; \{\, y \in Z \mid y{\uparrow}I = x{\uparrow}I \,\}$$

Note that $C_Z(x{\uparrow}I)$ is a $|I|$ dimensional subset of Z.

C. The Continuous Model

The following continuous version of the discrete model was introduced in [JordPR2]. Assume a continuous product form stationary distribution:

$$\pi(x) \;\equiv\; K \prod_{i=1}^{n} f_i(x_i) \qquad\quad f_i(x_i) \text{ continuous on } x_i \ge 0 \qquad (2)$$

on a state space given by $Ax \le b$. The state is x, a vector of length n of non-negative real numbers; A is a m×n matrix of non-negative real numbers representing resource usage for each service type; b is a vector of length m of positive real numbers representing total numbers of each resource type.

Assume that expectation is concave on cross sections:

P3:

For any two cross sections C_1 and C_2,

$$C_1 \equiv C_Z(x{\uparrow}I) \quad \& \quad C_2 \equiv C_Z((x+\beta e_j){\uparrow}I), \;\; j \notin I,$$

$$E_{[\alpha C_1 + (1-\alpha)C_2]} X_i \;\ge\; \alpha E_{[C_1]} X_i \;+\; (1-\alpha) E_{[C_2]} X_i \qquad \forall i \in I$$

$$\text{where } \alpha C_1 + (1-\alpha) C_2 \equiv C_Z((x+(1-a)\beta e_j){\uparrow}I)$$

D. The Optimal C.C. Policy

The optimal c.c. policy for the continuous model is obtained by repeatedly removing type "I" sets that produce below average revenue from the exterior of the state space. To define these sets, we introduce

$$M_I(R) \;\equiv\; \left(\, x \in Z \;\middle|\; r\!\left(E_{C_Z(x{\uparrow}I)}\; X\right) > R \,\right)$$

$M_I(R)$ is the set of states in "I" type cross-sections generating above average rates of revenue.

We wish to keep those states in all M_I. To do this we must also keep states "below" M_I, namely

$$BM_I(R) \equiv M_I(R) + ss(M_I(R))$$

$BM_I(R)$ is thus coordinate convex.

The optimal c.c. policy was characterized in [JordPR2]:

Theorem 1:

If X has a distribution on $Ax \le b$ given by (2) satisfying P3, then the optimal c.c. subset $Z^* \subseteq Z$ is defined by:

$$x \in Z^* \text{ iff } x \in Z \ \& \ x\uparrow I \in G_I^* \ \forall I \subset N \tag{3}$$

$$\text{where } G_I^* \equiv BM_I(R^*) \uparrow I \text{ are convex sets, and } R^* = E_{Z^*}\, r(X).$$

E. Algorithmic Implications

This characterization lends hope that for any reasonably sized discrete MSMR system, we may be able to find a *near optimal* control policy by approximating the discrete system by a continuous distribution, finding the optimal policy for the continuous model, and translating this back to the discrete domain. Hopefully, the trade-offs and trends observed in the continuous model hold in the discrete one.

Theorem1 suggests an algorithmic approach to finding the optimal (or near-optimal) control. The proof of theorem1 is constructive in nature and may be simpler to implement than methods of finding optimal or near-optimal policies directly in the discrete domain.

III. Algorithms

For comparison and experience, we implemented algorithms to find quantities of interest using both the discrete and continuous models. All of the results appearing in this paper, however, are based on the continuous model and theory.

All algorithms we implemented were written in a combination of S and C. S, a data analysis and graphics language, is good for writing high level algorithms and for drawing figures from data. C is required to obtain reasonable speed on low level algorithms. No attempt was made to optimize code; we only wished to demonstrate feasibility.

As expected, we found that the discrete algorithms were severely limited by the size of the state space they could handle. Throughputs, blocking probabilities, average revenue, and sensitivities were calculated by finding first and second moments and pertinent probabilities. The ability to calculate these moments and probabilities, however, is limited by the time to run through each state in the space, which is limited in turn by the number of states in the space. These routines quickly bog down when multiple sums over each state become a computational burden.

Optimal and near-optimal control policies, near-optimal resource acquisition, and optimal prices were calculated using both the moment routines and control policy routines. The performance of these were severely restricted by the brute force control policy routines we used which simply searched through all coordinate convex subsets of the state space. However, even if we had used the policy improvement schemes proposed in [Whit85] [KeyPr], we would have been restricted by the time to run through each state.

The continuous algorithms, on the other hand, are based on integration routines. A set of C routines calculates moments and pertinent probabilities, as before, and also finds the optimal control policy as outlined in theorem 1. Numerical integration is required in each. Higher level routines, in S, calculate similar results as do the discrete algorithms. Computation time is thus determined by the desired accuracy of the numerical integration instead of by the number of states.

IV. Moments

A. An Example

Consider a system with 2 services X & Y and 2 resources A & B. Assume service X occupies 2As and 1B and service Y occupies 1A and 2Bs. Assume there are 60 As and 60 Bs in the system. The state space for this system is shown in Figure 1. We will use this example in the remainder of this paper to illustrate algorithms to calculate quantities of interest.

The example was chosen to include a moderate number of states, so that moments could be calculated by either the discrete or continuous algorithms, but so that optimal control policies could only be calculated in reasonable time by the continuous algorithms.

B. Calculating Blocking Probabilities and Average Revenue

The majority of the work in the numerical integration routine is in calculating the density at each integration point. Therefore, rather than calling the integration routine separately to find each moment, we chose to have the integration routine keep one running sum for the normalization constant, plus one running sum for each first moment. One call thus returns all of the first moments. This suffices to find throughputs, blocking probabilities, and average revenue:

$$L_i = \mu_i \, E(X_i) \qquad P(F_i) = 1 - E(X_i) / \rho_i \qquad R = \sum_i r_i \, E(X_i)$$

If we chose to also calculate second moments, then we could have found sensitivities using these results and others from [Jord91]:

C. Varying the Load

To help illustrate the use of first moments and to demonstrate the need for control, consider our 2-service 2-resource example with r=(1,3) and with a load of ρ_Y=20. What happens as we vary ρ_X?

At low ρ_X, there is very little blocking of either service. As the load on service X starts to increase, EX increases with it, and EY decreases slightly since services X & Y are substitutes. As $P(B_Y)$ becomes significant, the trade-off becomes more severe and EY drops precipitously.

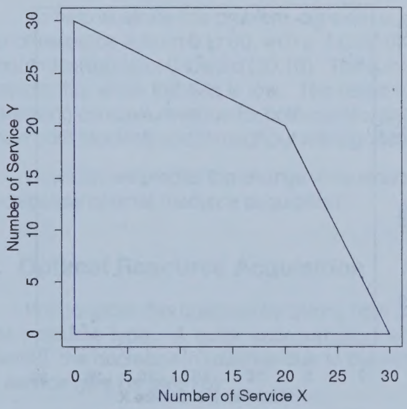


Figure 1 The State Space

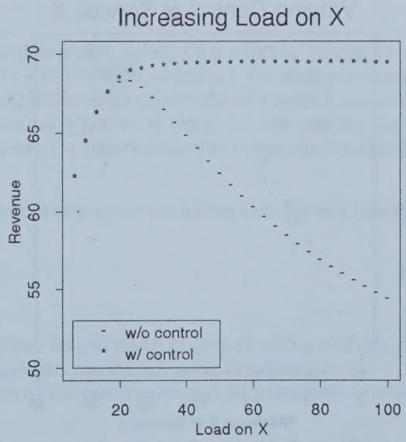


Figure 2 Increasing Load

Since $r_Y > r_X$, we should be concerned about this drop. Indeed, in the uncontrolled system, average revenue (Figure 2) increases only until $\rho_X \cong 23$ and then drops off with increasing ρ_X.

Allowing control of the system corrects this problem. Since $r_Y > r_X$, the optimal control for the continuous model is simply a restriction on the number of service X in the system. As the load increases, this restriction becomes more severe, eventually allowing only states with $X \leq 20$. As we would demand of the controlled system, revenue no longer dips at high ρ_X.

V. Control Policies

A. Calculating the Optimal Control Policy

Finding the optimal discrete c.c. policy involves looking at all coordinate convex subsets of the state space. This is computationally infeasible for anything but very small state spaces. The only alternatives proposed so far involve reducing the number of candidate control policies. Zachary uses a reduced-load approximation [Zach88] to treat the resources as independent. Key restricts his control policies to trunk reservation policies. Both use policy improvement to iteratively obtain better policies.

If our task, however, is to find the optimal c.c. policy for the *continuous* model (and use that to approximate the optimal discrete c.c. policy), then we can use theorem 1 to reduce the number of candidate control policies. We can restrict ourselves to looking among policies that produce a convex state space, and to restrictions as outlined in (3). Theorem1 implies that we should cut back on services of lesser value (the $x \uparrow \mathrm{I}$ services) to get increased throughput of some services of high value (the I services). More precisely, it states that if extreme points of the state space generate a below average rate of revenue, then we can do better by restricting the system from entering these regions. An algorithm can be written to iteratively place and increase restrictions at every such extreme point.

B. Varying Control

To help illustrate the use of such an algorithm, consider the same example with ρ_X fixed at 30. How sensitive is the system to the amount of control exercised? As we vary the maximum number of service X allowed, revenue moves as shown in Figure 3. The vertical difference at ρ_X=30 on Figure 2 shows up here as the difference between the peak value of R and the value at maximum X = 30. Here the peak is prominent and the control algorithm can find it easily in 3 iterations of the outer control loop.

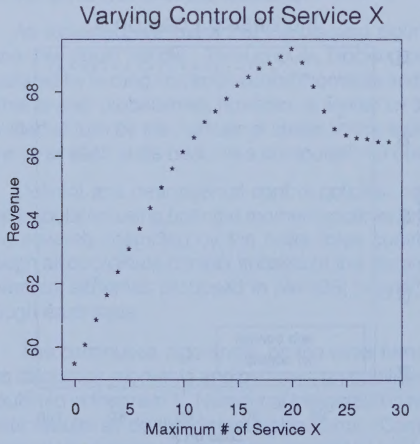


Figure 3 Varying Control: r=(1,3)

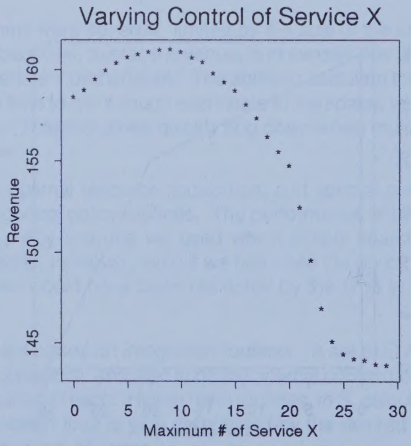


Figure 4 Varying Control: r=(1,8)

The optimal control point becomes much more sensitive at slightly higher r_Y/r_X. If we set r=(1,8), the peak is rounder (Figure 4). The optimal control is stronger, but is also tougher to find.

VI. Pricing

A. Setting Prices

Everything up to this point has assumed fixed prices, fixed demand and a static resource pool. We would like to consider strategies for setting prices and buying resources, from the point of view of a monopolist.

First consider the task of setting prices. A little calculus gives us:

$$\frac{\partial R}{\partial r_j} = \sum_i r_i \frac{\partial L_i}{\partial r_j} + L_j$$

So given a proposed set of prices, we can figure out how much would be gained or lost by a small hike in price if we know sensitivity of throughput to price. This part, however, can be expressed as:

$$\frac{\partial L_i}{\partial r_j} = \sum_k \frac{\partial L_i}{\partial \lambda_k} \frac{\partial \lambda_k}{\partial r_j}$$

The first piece, $\partial L_i/\partial \lambda_k$, is the sensitivity of throughput to demand, which we already discussed in section IV.B.. The second piece, $\partial \lambda_k/\partial r_j$, times r_j/λ_k is the elasticity of demand for service k to the price of service j.

We can therefore calculate optimal prices by iteration with a standard gradient algorithm, given a set of price elasticities and a numerical integration routine that returns first and second moments.

B. Buying Resources

Now consider the problem of acquiring resources. Assume you face fixed prices and demands, but can purchase resources at fixed prices to increase throughput by decreasing blocking. How many of each type of resource should you buy?

To help illustrate this problem, consider our example again. What is the effect of varying the number of resource A from 0 to 60, with ρ = (20,10) and r = (1,3)? Without control, the state increases in a smooth fashion from 0 toward (20,10). The control algorithm takes advantage of excess X capacity and restricts this when the #As is low. The result in revenue is shown in Figure 5. We see the expected increasing concave revenue for both curves, and we see the benefit of control in the intermediate range when both blocking and throughput are significant.

Now can we predict the change in revenue due to a change in resources and use this information to calculate optimal resource acquisition?

C. Optimal Resource Acquisition

We consider this question by asking how much can be gained by buying or selling one of a particular resource type. A quick approximation is suggested by the following observation due to Kelly [Kell88]: the decrease in revenue due to the removal of *the batch of resources necessary to offer one of service type i* is given by

$$\Delta R = \frac{\mu_i}{Ex_i} \frac{\partial R}{\partial \mu_i}$$

Since $\partial R/\partial \mu_i$ can be found using sensitivities given by first and second moments, the calculation of this quantity is simple.

To find the decrease in revenue due to the removal of a *single* resource requires a little more work. Removing a removable set A from Z results in a change in revenue of

$$\Delta R = \frac{\pi(A)}{1 - \pi(A)} (E_A R - ER)$$

So removing one of a resource results in a change in revenue given by the above equation with $\pi(A)$ = the probability that all resources of that type are in use, and with $E_A R$ = the average rate of revenue while all resources of that type are in use. This requires a one dimensional integration along each upper boundary of the state space.

Now since buying resources should result in a roughly increasing concave revenue, one strategy might be to repeatedly calculate the above ΔR's, and choose which resource to buy, given costs of resources, to maximize profit, until marginal revenue equals marginal cost. This should result in a operating curve of revenue versus cost a little bit off of the optimal randomized policy revenue versus cost curve.

To illustrate, we ran the suggested resource purchasing algorithm on this sample system. We started with just 6 of each resource, and assumed each costs 1. With a load of $\rho = (20,10)$, we let the algorithm choose which resource to buy at each point, based on expected gains in revenue and costs for each resource acquisition. With revenues of $r=(1,3)$, optimal resource acquisition produced revenue vs. cost curves as shown in Figure 6. As expected, this graph exhibits the concavity upon which we can find the optimal marginal revenue equals marginal cost point, and the benefit of control in the intermediate range when both blocking and throughput are significant.

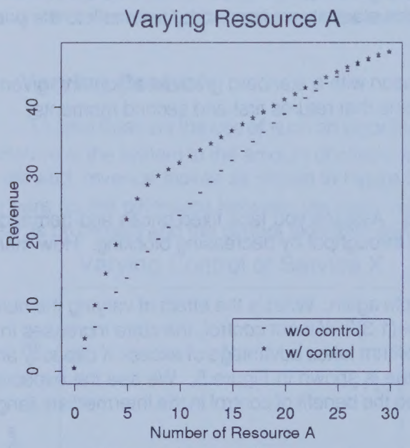


Figure 5 Varying Resource A

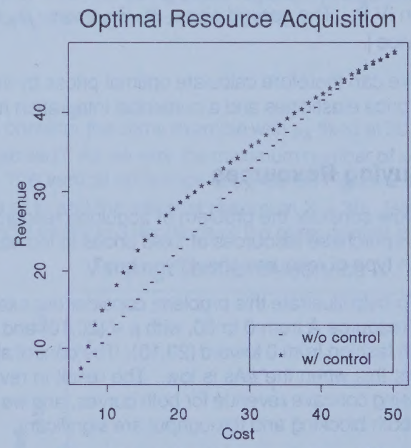


Figure 6 Optimal Resource Acquisition

VII. Parting Thoughts

In this paper, we have demonstrated the feasibility and computational advantages of using a continuous model of a multiple service, multiple resource system to calculate quantities of interest such as throughputs, blocking probabilities, average revenue, sensitivities, optimal and near-optimal control policies, near-optimal resource acquisition, and optimal prices.

This analysis, however, is limited by the requirement that resource usage is simultaneous. A service may not need two types of resources simultaneously, but first one and then the other. This latter type of sharing can not be addressed by the models in this paper, but instead require a model cognizant of changes in time.

Appendix 1. References

[Aein78] J. M. Aein, "A multi-user-class, blocked-calls-cleared, demand access model", *IEEE Transactions on Communications*, Vol. 26 No. 3, pp. 378-385, March 1978.

[Burm84] D. Y. Burman, J. P. Lehoczky, and Y. Lim, "Insensitivity of blocking probabilities in a circuit-switched network", *Journal of Applied Probability,* Vol. 21, pp. 850-859, 1984.

[Fosc83] G. J. Foschini and B. Gopinath, "Sharing memory optimally", *IEEE Transactions on Communications,* Vol. 31 No. 3, pp. 352-360, March 1983.

[Jord91] S. Jordan and P. Varaiya, "Throughput in Multiple Service, Multiple Resource Communications Networks", *IEEE Transactions on Communications,* Vol. 39 No. 8, August 1991.

[JordPR1] S. Jordan and P. Varaiya, "Control of Multiple Service, Multiple Resource Communications Networks", *submitted to IEEE Transactions on Communications.*

[JordPR2] S. Jordan and P. Varaiya, "A Continuous Model of Multiple Service, Multiple Resource Communications Networks", *submitted to IEEE Transactions on Communications.*

[Kauf81] J. S. Kaufman, "Blocking in a shared resource environment", *IEEE Transactions on Communications,* Vol. 29 No. 10, pp. 1474-1481, October 1981.

[Kell86] F. P. Kelly, "Blocking probabilities in large circuit-switched networks", *Advances in Applied Probability,* Vol. 18, pp. 473-505, 1986.

[Kell88] F. P. Kelly, "Routing in circuit-switched networks: optimization, shadow prices and decentralization", *Advances in Applied Probability,* Vol. 20, pp. 112-144, 1988.

[KeyPr] P. B. Key, "Optimal control and trunk reservation in loss networks", *preprint,* 1989.

[Krai84] B. Kraimeche and M. Schwartz, "Circuit access control strategies in integrated digital networks", *Proceedings of the IEEE Conference on Information Systems,* pp. 320-235, 1984.

[Mitr87] D. Mitra, "Asymptotic analysis and computational methods for a class of simple, circuit-switched networks with blocking", *Advances in Applied Probability,* Vol. 19, pp. 219-239, 1987.

[NainPr] P. Nain, "Qualitative properties of the Erlang blocking model with heterogeneous user requirements", *preprint,* January 1989.

[Ross89a] K. W. Ross and D. Tsang, "Optimal Circuit Access Policies in an ISDN Environment: A Markov Decision Approach", *IEEE Transactions on Communications,* Vol. 37 No. 9, pp. 934-939, September 1989.

[Ross89b] K. W. Ross and D. Tsang, "The Stochastic Knapsack Problem", *IEEE Transactions on Communications,* Vol. 37 No. 7, pp. 740-747, July 1989.

[RossPr] K. W. Ross and D. D. Yao, "Monotonicity properties for the stochastic knapsack", *preprint,* December 1988.

[Souz88] E. Souza E Silva and R. R. Muntz, "Simple relationships among moments of queue lengths in product form queueing networks", *IEEE Transactions on Computers,* Vol. 37 no. 9, pp. 1125-1129, Sept. 1988.

[Tsan90] D. Tsang and K. W. Ross, "Algorithms to determine exact blocking probabilities for multirate tree networks", *IEEE Transactions on Communications,* Vol. 38 No. 8, pp. 1266-1271, August 1990.

[Virt88] J. T. Virtamo, "Reciprocity of blocking probabilities in multiservice loss systems", *IEEE Transactions on Communications,* Vol. 36 No. 10, pp. 1174-1175, October 1988.

[Whit85] W. Whitt, "Blocking when service is required from several facilities simultaneously", *AT&T Technical Journal,* Vol. 64, pp. 1807-1856, 1985.

[Zach88] S. Zachary, "Control of stochastic loss networks, with applications", *Journal of the Royal Statistical Society B,* Vol. 50 No. 1, pp. 61-73, 1988.

Optimality of Index Policies for Stochastic Scheduling Problems with Switching Penalties

Mark P. Van Oyen, Dimitrios G. Pandelis, and Demosthenis Teneketzis

Dept. of Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, MI 48109-2122

Abstract: We present the main results of an investigation of the impact of switching penalties on the nature of optimal scheduling policies for systems of parallel queues without arrivals. We prove that optimal scheduling policies are characterized by an index rule for two types of switching penalties incurred when switching among queues: lump sum costs and time delays.

1. Introduction

We consider the stochastic sheduling of a single server in a system of N parallel queues without arrivals. The objective is to characterize a non-preemptive, non-idling scheduling policy which minimizes the sum of an expected discounted holding cost and an expected discounted switching penalty (switching cost or switching delay) incurred whenever the server moves from one queue to another.

The novel feature of this work is the determination of *optimal* stochastic scheduling policies when penalties are incurred for changes in resource allocation. Whereas stochastic scheduling problems without switching cost have received much attention (see for example Gittins (1989); Walrand (1988); and references therein), there is little work on optimal scheduling and resource allocation with switching penalties (see Glazebrook (1980); Gupta, Gerchak, and Buzacott (1987); and Hofri and Ross (1987)). The reason, it appears, is the difficulty of such problems.

One feature of many applications that is notably absent from our formulation is that of job arrivals. The problem with arrivals and switching penalties is difficult and researchers have only *partially* characterized optimal policies for only *special cases of two parallel queues* (see Gupta, Gerchak, and Buzacott (1987) and Hofri and Ross (1987)). On the other hand, we derive *optimal* scheduling policies for a model of N parallel queues with switching penalties, linear holding costs, general service distributions and no arrivals. Thus, our work differs from that of Gupta, Gerchak, and Buzacott (1987) and Hofri and Ross (1987). It is also different from Glazebrook (1980) who identifies certain conditions on the service distribution for which existing algorithms can be used to solve stochastic scheduling problems with precedence constraints and switching costs associated with individual jobs.

The paper is organized as follows: We formulate the problem precisely in Section 2. In Section 3, we first prove that only exhaustive policies can be optimal, then we show that optimal policies are characterized by indices and calculate these indices for the problem with switching cost and with switching delay. Finally, we discuss in Section 4 several extensions of the model formulated in Section 2.

2. Problem Formulation

We consider a system of N queues (nodes) and a single server; alternatively, one may think of a single queue and server with N classes of jobs. We are concerned only with the scheduling of the jobs initially present in the system. The N queues are differentiated by service distribution and holding cost. The lengths of the service periods for all jobs are mutually independent random variables. The service period of a job of type n ($n = 1, 2, \ldots, N$) possesses a general distribution with mean $\frac{1}{\mu_n}$ (assume $0 < \frac{1}{\mu_n} < \infty$). For every job of type n, an instantaneous holding cost c_n is incurred until the instant the job leaves the system.

In addition to holding costs, we assess a penalty for switching. Two optimization problems are considered; one treats lump sum switching costs and the other switching delays. These penalties are defined as follows.

Switching Cost. A switching cost, K ($K > 0$), is incurred at each instant the server completes a job of a certain type and then chooses to serve a job of a different type.

Switching Delay. A random length of time, D, is required to switch from one queue to any other; thus, the holding cost incurred during the switching delay is the implicit penalty. The successive delays are nonnegative i.i.d. random variables with a general distribution and finite mean.

The set of admissible strategies is taken to be the class G of non-idling, non-preemptive, and non-anticipative scheduling policies. With $\mathbb{R}^+$ ($\mathbb{Z}^+$) denoting the non-negative reals (integers), let $\{X_n(t) : t \in \mathbb{R}^+\}$ be the right-continuous queue length process of node $n \in \{1, 2, \ldots, N\}$ at time t (we include the node n customer, if any, in service at time t). We denote the vector of initial queue lengths by $x = (x_1, x_2, \ldots, x_N) = X(0^-) \in (\mathbb{Z}^+)^N$. Let $n^g(t)$ be a right-continuous process which describes the location of the server at time t under policy g. For both the problem of switching cost and switching delay, the objective is to characterize a policy which minimizes the infinite horizon expected discounted cost. Specifically, let $\Gamma^g = \left\{t \in \mathbb{R}^+ : n^g(t^-) \neq n^g(t)\right\}$ be the set of random switching instants under policy g; we see that Γ^g contains at least $N-1$ elements and at most $(\sum_n x_n - 1)$. For the problem of switching costs, the expected α-discounted cost $J^g(x)$ due to policy $g \in G$ is defined as

$$J^g(x) \triangleq E\left\{\int_0^\infty e^{-\alpha t} \sum_{n=1}^{N} c_n X_n^g(t)dt + K \sum_{t\in\Gamma^g} e^{-\alpha t} \middle| X(0^-) = x\right\} , \tag{2.1}$$

The objective is to characterize a policy g^* such that

$$J^{g^*}(x) = \min_{g \in G} J^g(x) \ . \tag{2.2}$$

In the case of switching delay, the cost to be minimized is

$$\overline{J}^g(x) \triangleq E\left\{\int_0^\infty e^{-\alpha t} c^g(t) dt \middle| X(0^-) = x\right\}. \tag{2.3}$$

Without loss of optimality, the minimization of both problems can be restricted to the class $G^{PM} \subset G$ of pure Markov policies.

For simplicity of notation, in the sequel we will omit the dependence of the cost on x, thus denoting $J^g(x)$ and $\overline{J}^g(x)$ simply by J^g and $\overline{J}^g$ since x is fixed.

3. Main Results

In the case of zero switching cost and no switching delay, it is known (Proposition 8.2.2 of Walrand (1988)) that the policy that serves jobs in decreasing order of $c_i \mu_i$ minimizes the total expected holding cost. Such a "$c\mu$-rule" that serves each queue in an exhaustive manner incurs $(N-1)$ switches. Since every policy requires at least $(N-1)$ switches, it follows that in the case of switching cost when $\alpha = 0$, it is optimal to serve the nodes exhaustively in decreasing order of $c_i \mu_i$.

When $\alpha > 0$ in either problem, or when $\alpha = 0$ in the case of switching delay, different switching times result in different switching penalties and it is no longer clear that a $c\mu$-type rule is optimal. Thus, we concentrate on the scheduling problem of switching delays with discount parameter $\alpha \geq 0$ and on the switching cost problem with $\alpha > 0$. We begin with a fundamental structural property characterizing an optimal policy for either problem. First we need the following:

Definition 1: A policy g is said to be *exhaustive* if according to g the server never leaves a node before completing the service of all the customers at that node.

Theorem 1: Only exhaustive policies can be optimal.

For a proof of Theorem 1, see Van Oyen, Pandelis, and Teneketzis (1991). As a consequence of Theorem 1, an optimal policy prescribes exactly $N-1$ switches. Theorem 1 provides a reduction in the class of policies which are candidate solutions to the optimization problem. The number of policies in G^{PM} is large and depends on the initial queue length. Theorem 1 limits the search to the set G^E of exhaustive pure Markov policies; G^E contains only $N!$ policies. We now proceed to further characterize an optimal policy by an index rule.

3.1. Switching Cost: $\alpha > 0$

For $g \in G^E$, we can rewrite (2.1) as

$$J^g = \sum_{n=1}^{N} c_n x_n \alpha^{-1} - K - \sum_{n=1}^{N} E\left\{\int_0^\infty e^{-\alpha t} c_n Y_n^g(t)dt - Ke^{-\alpha t^g(n)}\right\} , \tag{3.1}$$

where $(x_1, x_2, \ldots, x_N)$ are the initial queue lengths; $\{Y_n^g(t), t \geq 0\}$ is the process describing the number of departures from node n through time t; and $t^g(n)$ denotes the random time at which g initiates the service of node n. If node j is the first served under g, $t^g(j) = 0$ and $Ke^{-\alpha t^g(j)} = K$; thus the term $-K$ accounts for the fact that no switching cost is incurred upon entering the first node served. Because of Theorem 1, the minimization of J^g is equivalent to the maximization of R^g over the class of exhaustive policies G^E, where

$$R^g \triangleq \sum_{n=1}^{N} E\left\{\int_0^\infty e^{-\alpha t} c_n Y_n^g(t)dt - Ke^{-\alpha t^g(n)}\right\} . \tag{3.2}$$

Let σ_n denote a random service period for node n and $S_n \triangleq E\{e^{-\alpha \sigma_n}\}$. The i.i.d. nature of service periods in node n allows us to write R^g in the following form:

$$R^g = \sum_{n=1}^{N} E\left\{e^{-\alpha t^g(n)}\right\} \left(c_n \alpha^{-1} S_n (1 - S_n)^{-1} (1 - S_n^{x_n}) - K\right) . \tag{3.3}$$

The problem of maximizing R^g can be viewed as a special case of a multi-armed bandit problem with switching cost. Consistent with the interpretation of dynamic allocation indices as measures of reward rate, we define the index of queue i $(i = 1, 2, \ldots, N)$ as

$$\nu_i = \frac{E\{\int_0^{\tau_i} e^{-\alpha t} c_i S_i (1 - S_i)^{-1} dt \; - K\}}{E\{\int_0^{\tau_i} e^{-\alpha t} dt\}} , \tag{3.4}$$

$$= c_i S_i (1 - S_i)^{-1} - \alpha K (1 - S_i^{x_i})^{-1} . \tag{3.5}$$

where τ_i is the length of time required to clear queue i (that is, the sum of x_i service periods of queue i). The following result shows that the indices defined above can be used to determine an optimal policy.

Theorem 2: An index rule is optimal for the problem with switching cost: serve the queues exhaustively in decreasing order of the dynamic allocation indices ν_i.

For a proof of this result see Van Oyen, Pandelis, and Teneketzis (1991). For fixed α, the index of each node reveals the tradeoff between the initial queue length, the switching cost, and the expected discounted reward rate (i.e. $cS(1 - S)^{-1}$) received by serving a given queue. The longer the queue, the less is the penalty incurred due to switching. Hence, higher priority should be given to queues with a sufficiently large number of customers and high $cS(1-S)^{-1}$, so that the overall reward received by serving the queue minus the switching cost is maximized. If $K=0$, Theorem 2 gives a generalized version of the $c\mu$-rule. On the other hand, if the holding costs and service distributions are the same for all queues, the above discussion indicates that the optimal policy should proceed in order of decreasing queue length.

Remark: If the service period distribution of node i is exponential with rate μ_i, then $S_i = \mu_i(\alpha+\mu_i)^{-1}$ and (3.5) gives

$$\nu_i = c_i\mu_i\alpha^{-1} - \frac{\alpha K}{1-(\mu_i(\alpha+\mu_i)^{-1})^{x_i}} \,. \tag{3.6}$$

3.2. Switching Delay: $\alpha = 0$

We begin the analysis of the problem with delay by considering the undiscounted case. The notation developed in the previous sections is used here. We write the expected cost of policy $g \in G^E$ given initial state $(x_1, x_2, \ldots, x_N)$ as

$$\overline{J}^g = E\left\{\int_0^\infty \sum_{n=1}^{N} c_n(x_n - Y_n^g(t))dt\right\} \,. \tag{3.7}$$

Since every job has a finite mean service period and only nonidling policies are considered, $\overline{J}^g$ is finite. Let $t^g(n)$ denote the instant at which g initiates the service of node n (the switching delay prior to serving n is included). A brief computation reveals that

$$\overline{J}^g = \sum_{n=1}^{N}\left(c_n x_n E\{t^g(n)\} + c_n\mu_n^{-1}x_n(x_n+1)/2\right) . \tag{3.8}$$

Define the dynamic allocation index of node n as

$$\begin{aligned}\overline{\nu}_n &= \frac{c_n x_n}{E\{\tau_n + D\}} \\ &= c_n\mu_n\frac{x_n\mu_n^{-1}}{x_n\mu_n^{-1}+E\{D\}} \,.\end{aligned} \tag{3.9}$$

Then, the optimal scheduling policy is characterized as follows:

Theorem 3: For the problem with switching delay, it is optimal to serve the nodes exhaustively in decreasing order of $\overline{\nu}_n$.

See Van Oyen, Pandelis, and Teneketzis (1991) for a proof of the theorem.

3.3. Switching Delay: $\alpha > 0$

With $\alpha > 0$, the analysis of the problem with switching delay closely follows that of Section 3.1. For $g \in G^E$, we find

$$\overline{J}^g = \sum_{n=1}^{N} c_n x_n \alpha^{-1} - \overline{R}^g \,,$$

where

$$\overline{R}^g = \sum_{n=1}^{N} E\left\{e^{-\alpha t^g(n)}\right\} c_n S_n \alpha^{-1}(1-S_n)^{-1}(1-S_n^{x_n}) \,. \tag{3.10}$$

As in Section 3.2, $t^g(n)$ is taken to include any switching delays incurred prior to processing node n.

Define the index of node n as

$$\begin{aligned} \overline{\nu}_n &= \frac{E\left\{\int_0^{\tau_n} e^{-\alpha t} c_n S_n (1-S_n)^{-1} dt\right\}}{E\left\{\int_0^{\tau_n + D} e^{-\alpha t} dt\right\}} \\ &= \frac{c_n S_n (1-S_n)^{-1}(1-S_n^{x_n})}{1 - S_n^{x_n} E\{e^{-\alpha D}\}} \,. \end{aligned} \tag{3.11}$$

Theorem 3 remains valid for $\alpha > 0$ using the indices of (3.11).

Remark: In the case of exponential service distributions, the indices become

$$\overline{\nu}_n = c_n \mu_n \alpha^{-1} \frac{1 - (\mu_n/(\alpha+\mu_n))^{x_n}}{1 - (\mu_n/(\alpha+\mu_n))^{x_n} E\{e^{-\alpha D}\}} \,. \tag{3.12}$$

4. Extensions

Although a closed-form solution to the problem formulated in Section 2 has been obtained, there are many features of interest that are not contained in that formulation. One natural and important extension of the formulation would be the inclusion of job arrivals. Unfortunately, our analysis sheds little light on this issue since even in the case of deterministic job arrival instances, the exhaustive property of Theorem 1 does not apply. We suggest, however, that the form of the indices in the problem without arrivals may be useful as a basis for heuristic strategies for problems with arrivals and switching penalties.

Another extension of importance is that of multiple servers attending the N queues. The following example is illustrative of the subtleties encountered even in the case of equal deterministic service periods.

Example: Consider two servers, two queues, and deterministic one-unit service periods in both nodes. Assume a switching cost K (no delay). Both nodes have holding cost rates of one unit per job per unit time. Let the initial queue lengths be $x_1 = 8$ and $x_2 = 2$. For α such that $e^{-\alpha} < 1/2$, an exhaustive policy is optimal: each server processes four jobs of queue one, then switches to serve one job of queue two. For $1/2 \le e^{-\alpha} \le 1$, however, the optimal policy does not serve node one exhaustively: one server processes five jobs of queue one, the other three jobs of node one followed by both jobs in node two.

In the remainder of this section, we briefly discuss variations of the single-server problem formulated in Section 2. First we consider the scheduling problem formulated in Section 2 and we associate with each node n a set-up cost K^n incurred upon initiation of jobs of queue n, and a shut-down cost K_n incurred upon switching from node n to any other node. The optimal policy is again given by an index rule. The index of node i $(i = 1, 2, \ldots, N)$ is given by

$$\nu_i = c_i S_i (1 - S_i)^{-1} - \alpha (K^i + S_i^{x_i} K_i)(1 - S_i^{x_i})^{-1} \,. \tag{4.1}$$

Similar results hold when instead of a switching cost we associate with each node n set-up and shut-down delays D^n and D_n respectively. The optimal policy is again given by an index rule. For the cases $\alpha = 0$ and $\alpha > 0$ respectively, the index of node n $(n = 1, 2, \ldots, N)$ is calculated by

$$\overline{\nu}_n = c_n \mu_n \frac{E\{\tau_n\}}{E\{\tau_n + D^n + D_n\}} \,, \tag{4.2}$$

where τ_n is the total time required to serve x_n jobs in node n, and

$$\overline{\nu}_n = \frac{E\{e^{-\alpha D^n}\} c_n S_n (1 - S_n)^{-1} (1 - S_n^{x_n})}{1 - S_n^{x_n} E\{e^{-\alpha D^n}\} E\{e^{-\alpha D_n}\}} \,. \tag{4.3}$$

The optimality of an index rule rests on the assumption that the switching penalty associated with serving all the jobs of a given queue is not dependent on the sequence of past or future queues served. A natural generalization of the switching penalty model of Section 2 is to consider a transition-dependent switching cost K_{ij} (or delay D_{ij}) associated with a switch from a job of node i to one of node j. Although an index rule of the previous form is not optimal, it can be shown under a reasonable assumption that there is an exhaustive policy which is optimal. Assume for all $i, l, j \in \{1, 2, \ldots, N\}$ that

$$K_{ij} \leq K_{il} + K_{lj} \tag{4.4}$$

in the problem with switching costs, or

$$E\{e^{-\alpha D_{ij}}\} \geq E\{e^{-\alpha (D_{il} + D_{lj})}\} \tag{4.5}$$

in the problem with switching delays. Note that (4.5) is implied by the assumption that D_{ij} is stochastically less than or equal to $D_{il} + D_{lj}$. Under either (4.4) or (4.5), the proof of Theorem 1 holds and proves that exhaustive policies are optimal for problems with either switching cost or delay. It remains to search G^E (which contains $N!$ list policies) to determine the optimal policy.

5. Conclusions

We have investigated the impact of switching penalties on the nature of optimal scheduling policies for systems of parallel queues without arrivals. The results clearly show the role of the initial queue lengths, the reward rate (e.g. $c\mu\alpha^{-1}$ or $cS(1 - S)^{-1}$), and the switching penalty in determining the optimal schedule.

Acknowledgement

The work of Mark Van Oyen was supported by a Regents Fellowship from the University of Michigan.

References

[1] Gittins, J.C. (1989) *Multi-armed Bandit Allocation Indices* Wiley, New York.

[2] Glazebrook, K.D. (1980) On stochastic scheduling with precedence relations and switching cost, *J. Appl. Prob.* **17**, 1016–1024.

[3] Gupta, D., Gerchak, Y., and Buzacott, J.A. (1987) On optimal priority rules for queues with switchover costs, Preprint.

[4] Hofri, M. and Ross, K.W. (1987) On the optimal control of two queues with server setup times and its analysis, *SIAM J. Comput.* **16**, 399–420.

[5] Van Oyen, M.P., Pandelis, D., and D. Teneketzis (1991) Optimality of Index Policies in Stochastic Scheduling Problems with Switching Penalties, Technical Report No. 276, Communications and Signal Processing Laboratory of the University of Michigan, Ann Arbor.

[6] Walrand, J. (1988) *An Introduction to Queueing Networks* Prentice Hall, Englewood Cliffs.

BURST-REDUCTION PROPERTIES OF LEAKY BUCKET STRATEGIES IN ATM NETWORKS

Venkatachalam Anantharam
Cornell University
and
Panagiotis Konstantopoulos
University of California, Berkeley

Abstract

The leaky bucket is a simple flow control scheme for ATM networks. An arriving cell can be transmitted only if it finds a counter in the counter buffer, in which case it is transmitted instantaneously by consuming a counter. Else, if the counter buffer is empty, the cell has to wait until the generation of a new counter. The control parameter is the counter buffer size C. We examine the burstiness of the output flow as a function of C and show that the burstiness increases with C. As a consequence, the output flow is always less bursty than the input flow. Our result is true for fairly arbitrary input flows and deterministic counter generation times.

1 Introduction

An ATM (Asynchronous Transfer Mode) network is designed so that it supports various types of asynchronous traffic with widely different rates (e.g. video, audio, HDTV, etc.). This clearly creates a burden for the switching elements of the network and introduces new problems that are not encountered in conventional packet switching networks. It is thus desirable to have some kind of control protocol that will somewhat alleviate the switches from the burden of having to switch very bursty traffic. This protocol should be fast and easy to implement and capable to operate in real time. One of the proposed standards is the so-called *leaky bucket* control scheme.

The leaky bucket operates as follows: The arrival process that consists of packets of rather small and fixed size, known as *cells* is controlled by *counters* that are stored in a buffer of fixed size C. An arriving cell can be transmitted only if it finds a counter in the counter buffer; in which case the cell is transmitted without delay and the contents of the counter buffer are decreased by 1. If the counter buffer is empty then the arriving cells have to be temporarily stored in a buffer. Counters are generated at a fixed rate. A stored cell can be transmitted immediately upon the generation of a new counter.

The leaky bucket scheme and variants of it have recently generated considerable interest in the literature. See for example Eckberg et al [5]. The reason is that the leaky bucket type of schemes are easily implementable and there are reasons to believe that they perform better than other flow control mechanisms. For simplified studies of this comparison see Rathgeb [7] and Dittman et al [4]. These papers appeared in the recent IEEE journal on selected areas in communications. In fact, the whole journal was devoted in the teletraffic analysis of ATM systems. See also Buttó et al [3] for a fluid model of the leaky bucket and Berger [2] for a performance analysis of a related model. More references can be found in all these papers [2], [3], [4], [5], [7].

In this paper we present a stochastic model for the leaky bucket protocol that captures the dynamics of the system and that is applicable to fairly arbitrary cell arrival and counter generation processes. We model the cell arrival process and the token arrival process as general jointly stationary and ergodic processes. We first show that the state process of the bucket admits a stationary and ergodic regime that is unique. In section 3 we see how, from the state process, we can construct the cell departure process (that is, the process of actual transmission times) from the leaky bucket, for various values of C. A notion of ordering between these processes is also introduced. In section 4, we sketch the proof of the fact that the leaky bucket scheme is *burst reducing* and that the burst reduction increases as C decreases.

2 State process and stationary regime

Let $x(t)$ denote the number of cells in the cell buffer at time t. The cell buffer is of unlimited size. Let $y(t)$ denote the number of counters in the counter buffer at time t. The counter buffer has capacity C. Thus $0 \leq y(t) \leq C$. From the description of the leaky bucket protocol of section 1 we see that a situation where both x and y are nonzero is only a transient one. That is, even if initially we have $x_0 > 0$ cells and $y_0 > 0$ counters, after at most y_0 transmissions one of the variables x or y will be zero. Thus we may assume that $x(t)y(t) = 0$ for all t. Consequently, the system can be described by $z(t) = x(t) - y(t)$, the difference between the contents of the two buffers. So, if $z(t) > 0$ there are $z(t)$ cells and no counters, if $-C \leq z(t) < 0$ there are $-z(t)$ counters and no cells, and if $z(t) = 0$ there are neither cells nor counters. We will from now on assume that $\{x(t)\}$, $\{y(t)\}$, and any other function of time is left-continuous. Assume that the cells arrive at times $t_i, i = 1, 2, \ldots$ and that counters are generated at times $s_j, j = 1, 2, \ldots$. Both $\{t_i\}$ and $\{s_j\}$ are increasing sequences, and $t_1 \geq 0$, $s_1 \geq 0$. If a counter happens to be generated at a time t when the counter buffer is full (that is, according to our notation, $z(t) = -C$) then the counter is considered lost. We make the convention that if a cell arrives at the time a counter is generated, it is as if the cell arrived immediately after the counter. Denote now by z_n the value of the process z immediately before time s_n (generation of n-th counter). Let $A(I) = \sum_i 1\{t_i \in I\}$ denote the number of arriving cells during the interval I and let $a_n = A[s_{n-1}, s_n)$ be the number of cells arriving into the leaky bucket on the interval $[s_{n-1}, s_n)$. We can now see that z_n is given by the following recursion

$$z_{n+1} = z_n + a_{n+1} - 1\{z_n - 1 \geq -C\}. \tag{1}$$

Indeed, if $z_n \geq 0$ there are z_n cells immediately before s_n. This number is decreased by 1 (if $z_n = 0$ this means that the counter that arrives at time s_n will be stored in the buffer) and, since no counter is generated on (s_n, s_{n+1}) it is subsequently increased by a_n in order to give z_{n+1}. Similarly, if $z_n < 0$ there are $-z_n$ counters and no cells immediately before s_n. If $-z_n > C$ the counter buffer is not full, so the counter arriving at s_n is not lost and z_n is decreased by 1. It is subsequently increased by a_n in order to give z_{n+1}.

Note that, in order to derive equation (1) we did take into account the aforementioned convention concerning the conflict between a cell and a counter arrival. In practice the software is written so that only one event occurs at a particular instance of time.

Define now the quantity

$$w(t) = z(t) + C = x(t) - y(t) + C \, .$$

The reason for this definition will be made clear below. Let w_n be the value of w just before s_n. By adding the constant C to both sides of the recursion (1) for z_n we readily get the following recursion for w_n:

$$\begin{aligned} w_{n+1} &= w_n + a_{n+1} - 1(w_n > 0) \\ &= (w_n - 1)^+ + a_{n+1} \, , \end{aligned} \tag{2}$$

where $x^+ = \max\{x, 0\}$. Note that w takes values in the set $\mathbf{Z}_+$ of nonnegative integers.

Let us now think of the cell arrival times $\{t_i\}$ and the counter generation times $\{s_j\}$ as doubly infinite random sequences of points over the real line, being jointly stationary and ergodic (under arbitrary time shifts) under some probability measure P. This assumption is fairly general and covers most of the real cases (ergodicity is an inherent property of nature, as well as stationarity–at least asymptotically). We denote by λ the rate at which cells arrive and we assume, without loss of generality, that the rate at which counters are generated is 1. The sequence $\{a_n\}$ is then a stationary and ergodic sequence of random variables, conditional on the event that there is a counter generation at the origin of time.

Note that, strictly speaking, one has to define a new probability measure P^0, called the Palm transformation of P with respect to the point process of counter generation times. The probability $P^0(S)$ of an event S can be interpreted as P-probability of S conditional on the event that there is a counter generation at the origin of time. It is P^0 (and not P) under which the random sequence $\{a_n, n \in \mathbf{Z}\}$ is stationary and ergodic. We will not insist on this technical point further.

We are now ready to state and prove the main theorem of this section:

Theorem 1 *If $\lambda < 1$ there exists a random process that satisfies the recursion (2) that is stationary and ergodic. Furthermore this process is unique. If $\lambda > 1$ then, with probability 1, w_n converges to ∞ as $n \to \infty$.*

Proof

Observing that the right hand side of the recursion (2) is nondecreasing in w_n, one realizes that Loynes' method (see [1], [6]) is applicable. For $m < n$ define

$$S_{m,n} = a_{m+1} + \ldots + a_n - (n - 1 - m) \, .$$

Assume that we start the recursion (2) at m with $w_m = 0$. Iterating forward, we can obtain the value at n. This will be denoted by $w_{m,n}$ and can be seen that is given by

$$w_{m,n} = \max_{m \le l \le n-1} S_{l,n} \, .$$

It is clear that $w_{m,n}$ increases as m decreases. We let

$$\tilde{w}_n = \lim_{m \to -\infty} w_{m,n} = \sup_{m<n} S_{m,n} \, . \tag{3}$$

By the law of large numbers, $\lim_{m\to\infty} S_{m,n}/(n-m) = \lambda - 1$, with probability 1. Indeed, the expectation of a_n is the ratio of the arrival rate λ over the counter generation rate. If we assume that $\lambda < 1$, the above observation implies (see expression (3)) that $\tilde{w}_n$ is finite with probability 1. It is also easy to see that $\{\tilde{w}_n\}$ satisfies the recursion (2) and that it is a stationary and ergodic random sequence. Thus, the first part of the theorem is proved. The second part follows easily from the observation that the law of large numbers $\lim_{n\to\infty} S_{0,n}/n = \lambda - 1$ holds and $\lambda - 1$ is now positive.

We finish this section with a few observations and remarks.

1. First, observe that the extension of the discrete time stationary process $\{w_n\}$ can be easily extended to give a continuous time stationary and ergodic process $\{w(t); t \in \mathbf{R}\}$.

2. Second, we notice that the distribution of w does not depend on the choice of the counter buffer size C. Nevertheless, the distributions of $z = w - C$, $x = \max\{w - C, 0\}$, and $y = \max\{C - w, 0\}$ do depend on C–as they should!

3. In particular, the probability that the counter buffer is full (either at an arbitrary time or at the time that a cell arrives) is equal to the probability that w is zero; hence it does not depend on C.

4. In practice, the counter generation times in the leaky bucket protocol are deterministic and equally spaced in time. Also, one usually assumes that the cell arrival process is Poisson. In this case the sequence $\{w_n\}$ is a Markov chain–in fact a reflected random walk whose stationary distribution can, at least in principle, be computed. It is clear that our model does include these special cases.

3 The leaky bucket departure process

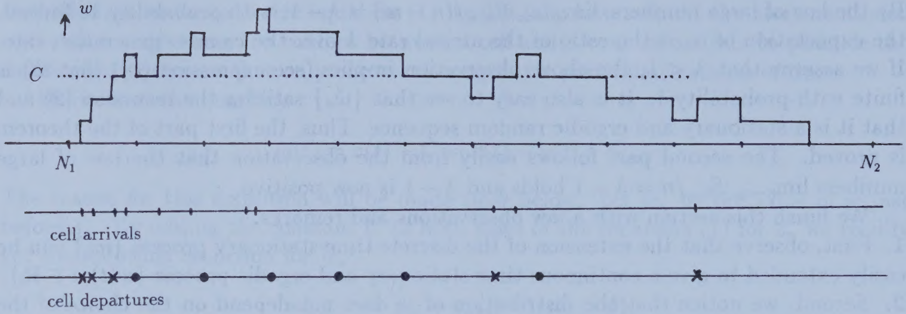
The departure process D^C from the leaky bucket with counter buffer capacity C is the point process of (random) times at which cells are transmitted. Thus $D^C(I)$ denotes the number of cells departing from the leaky bucket during the interval I. We start by introducing an ordering between the departure processes corresponding to different values of C. Recall that the cell arrival rate is λ and the counter generation rate is 1. From now on we will assume (see theorem 1) that $\lambda < 1$.

Definition 1 *We say that D^{C_1} is* less bursty *than D^{C_2} if and only if the steady state queue length in a deterministic single server queue with rate $\mu > 1$ and arrival process D^{C_1} is* stochastically smaller *than the steady state queue length in the same queue but with arrival process D^{C_2}.*

This definition is supposed to formalize the intuitive idea of "burstiness", at least in our specific case. Note that the deterministic service rate μ models the fact that cells are of fixed size.

The computation of the distribution of this steady state delay is a hard problem, even in the case when the cell arrival process is Poisson and the counter generation process deterministic (see remark 4 in the end of section 2). We thus need another approach for the comparison of the departure processes.

From the discussion of section 2 we see that the departure process from the leaky bucket can be constructed from a sample path of the w-process as in the figure below:



Construction of the leaky bucket departure process

The transitions with arrows correspond to cell departures.
The × denote departure times that coincide with arrival times.
The • denote departure times of delayed cells.

A cell departs either when w makes a transition from a level $i < C$ to $i+1$ (in which case the cell is transmitted without delay) or when w makes a transition from a level $i > C$ to $i-1$ (in which case the cell has a nonzero delay). Any other transition of w does not yield a cell departure. What is more important about this picture is that the departure processes for a whole range of systems, corresponding to different values of C, are constructed simultaneously (on the same probability space).

In the following section we consider the case of a deterministic counter generation process (but arbitrary cell arrival process) and show that D^{C_1} is less bursty than D^{C_2} if $C_1 < C_2$.

4 Burst reduction properties of the leaky bucket

Suppose that counters arrive at times $0, 1, 2, \ldots$. Assume that the departure process is fed into a deterministic queue with service rate $\mu > 1$. Thus all service times are constant and equal to $\sigma = \mu^{-1}$. Note that the term "departing cell" refers to a cell that leaves the leaky bucket and joins, without delay, the deterministic queue. The main result can be formulated as follows:

Theorem 2 *The leaky bucket departure process D^{C_1} is less bursty (in the sense of definition) than D^{C_2} if $C_1 < C_2$.*

Proof
The main idea here is that the departure processes of the systems with parameters $C_1 = C > 0$ and $C_2 = C + 1$ can be constructed on the same probability space, simply

from a path of the process w. Consider then a stationary left-continuous version of the w process, as in theorem 2 and the remarks following it.

To talk about stationary queue length we further need to construct the stationary queue length process on the probability space on which w is constructed. Let $N_1, N_2, \ldots$ be the times at which counters are generated and lost. Due to the construction of section 3 these points can be taken to be the same in both systems. Thus,

$$\begin{aligned} N_1 &= \inf\{t = 0, 1, 2, \ldots : w(t) = 0\}, \\ N_{j+1} &= \inf\{t = 0, 1, 2, \ldots : w(t) = 0\}, \; j = 1, 2, \ldots \end{aligned} \tag{4}$$

We refer to the piece of the process w between N_1 and N_2 as the "first cycle", that between N_2 and N_3 as the "second cycle", and so on. For the proof of theorem 2 we need to construct a stationary version of the queue length process $\{q(t)\}$ together with $\{w(t)\}$. The basic observation is that, in steady state, the queue length is zero at time N_j. This is stated in the following lemma:

Lemma 1 *The stationary versions of the processes $\{q(t)\}$ and $\{w(t)\}$ are such that $q(N_i) = 0$, $i = 1, 2, \ldots$. Thus the stationary queue length is zero at the beginnings of the cycles of w.*

The proof of this lemma is basically a counting argument in combination with the fact that the service time $\sigma = \mu^{-1}$ is smaller than 1. If the cycle has length n (i.e., $N_2 - N_1 = n$) then there are exactly $n - 1$ cells that arrived and departed during this cycle. It can be shown that these departures do not occur too late during the cycle so that the deterministic server, who works faster than counters are generated, can process the cells before the end of the cycle.

We denote the dependence on C of the queue length by writing $q^C(t)$. Let $\{v^C(t)\}$ be the workload process in the deterministic queue. Note that from $v^C(t)$ we can determine $q^C(t)$ from the formula

$$q^C(t) = \left\lceil \frac{v^C(t)}{\sigma} \right\rceil,$$

where $\lceil x \rceil$ denotes the smallest integer k such that $k \geq x$. Thus it suffices to show that the steady state workload, call it v^C is stochastically increasing in C.

Let α be the expectation of $N_2 - N_1$ (conditional that N_1 is 0). Let $ET^C(l)$ be the mean time that the v^C spends below or at level l during the random interval $[N_1, N_2)$, i.e.

$$T^C(l) = \int_{N_1}^{N_2} 1\{v^C(t) \leq l\} dt.$$

The stationary distribution of v^C is then given by

$$P(v^C \leq l) = \frac{ET^C(l)}{\alpha}.$$

To prove that v^C is stochastically smaller than v^{C+1} we must prove that

$$P(v^C \leq l) \geq P(v^{C+1} \leq l), \text{ for all } l.$$

Note that α does not depend on C. It therefore suffices to show that

$$T^C(l) \geq T^{C+1}(l), \text{ for all } l. \tag{5}$$

Consider the first cycle of the process w, i.e. the piece of the process between N_1 and N_2. If w does not exceed C throughout the cycle duration then the C- and $C+1$-departures coincide and (5) is automatically true. So suppose that this is not the case.

Assume that the first cycle has length n: $N_1 = 0$ and $N_2 = n$. Define

$$\begin{aligned} t_a &= \inf\{0 < t \leq n : w(t-) = C, w(t+) = C+1\} \\ t_b &= \inf\{0 < t \leq n : w(t-) = C, w(t+) = C-1\}. \end{aligned}$$

Set $t_a^{(1)} = t_a$, $t_b^{(1)} = t_b$ and define recursively

$$\begin{aligned} t_a^{(i+1)} &= \inf\{t_a^{(i)} < t \leq n : w(t-) = C, w(t+) = C+1\} \\ t_b^{(i+1)} &= \inf\{t_b^{(i)} < t \leq n : w(t-) = C, w(t+) = C-1\}, \text{ for } i = 1, 2, \ldots, \end{aligned}$$

with the convention $\inf \emptyset = +\infty$. Thus $w(t) \geq C$ for $t \in S = \bigcup_{i \geq 1} [t_a^{(i)}, t_b^{(i)})$, while $w(t) \leq C$, otherwise. We claim that

Lemma 2

$$v^C(t) = v^{C+1}(t), \text{ for all } t \in [0, n) - S.$$

Proof of lemma 2 We first observe that the workload is the same in both systems just before t_a: $v^C(t_a-) = v^{C+1}(t_a-)$. It thus suffices to prove that $v^C(t_b-) = v^{C+1}(t_b-)$.

Note that:

1. There is at least one C-departure between t_a and t_b; if there are more than one departures then they are equally spaced with all interdeparture times equal to 1 (these times correspond to downward transitions of the process w from levels higher than C).

2. The $C+1$-departures between t_a and t_b occur at earlier times than the C-departures and are not necessarily equally spaced; in fact, the discrepancies between the C- and $C+1$-departures correspond to transitions of w either from C to $C+1$ or from $C+1$ to C.

3. The number of C-departures between t_a and t_b is exactly equal to $\lfloor t_b - t_a \rfloor$, for the latter quantity is the number of transitions of w from a level $i > C$ to $i-1$. (Note that $\lfloor x \rfloor$ denotes the largest integer k such that $x \geq k$.) The number of $C+1$-departures is also $\lfloor t_b - t_a \rfloor$ (indeed, for every $C \to C+1$ transition between t_a and t_b there must be a corresponding $C+1 \to C$ transition).

Now observe that the the work brought by these $\lfloor t_b - t_a \rfloor$ departures on the interval between t_a and t_b is $\lfloor t_b - t_a \rfloor \sigma \leq t_b - t_a$. From the observations above we see that, if the deterministic server gives priority to this work that was generated between t_a and t_b, then this work will finish before t_b. Hence we have $v^C(t_b-) = v^{C+1}(t_b-)$. Thus $v^C(t) = v^{C+1}(t)$ for $t \in [t_b^{(1)}, t_a^{(2)})$. Hence $v^C(t_a^{(2)}-) = v^{C+1}(t_a^{(2)}-)$. The argument can then be repeated. This finishes the proof of lemma 2.

We now proceed to show inequality (5): $T^C(l) \geq T^{C+1}(l)$. In view of lemma 2, it is enough to show that

$$\int_{t_a^{(i)}}^{t_b^{(i)}} 1\{v^C(t) \leq l\} dt \geq \int_{t_a^{(i)}}^{t_b^{(i)}} 1\{v^{C+1}(t) \leq l\} dt, \text{ for all } i \geq 1. \tag{6}$$

We do that for $i = 1$. Let t_0 be the first time after t_a that v^C becomes zero. It should be clear that $v^C(t) \leq v^{C+1}(t)$ for all $t \in [t_b, t_0)$. Thus

$$\int_{t_a}^{t_0} 1\{v^C(t) \leq l\} dt \geq \int_{t_a}^{t_0} 1\{v^{C+1}(t) \leq l\} dt\,.$$

On the other hand, on $[t_0, t_b)$ the number of $C+1$-departures is less than or equal to the number of C departures. A few moments of reflection show that this implies

$$\int_{t_0}^{t_b} 1\{v^C(t) \leq l\} dt \geq \int_{t_0}^{t_b} 1\{v^{C+1}(t) \leq l\} dt\,.$$

Combining the two displays above we see that inequality (6) holds for $i = 1$. Likewise, it holds for $i > 1$. This concludes the proof of theorem 2.

5 Final remarks

1. According to theorem 2 the least bursty departure process is the one that corresponds to $C = 1$. The most bursty departure process is the one that corresponds to $C = \infty$. But the latter is exactly a replica of the cell arrival process (see section 3). Thus the output process is always less bursty than the input process, in the sense of definition 2.
2. This burst reduction property of the leaky bucket has an obvious cost: The delay of a cell in the buffer of the leaky bucket increases as C decreases. This is immediate from Little's law and the fact the fraction of time that the w process spends above C increases as C decreases. Our result can then be translated into a design guideline for choosing the optimal bucket size subject to a constraint on the allowable mean delay in the bucket control mechanism.

References

[1] Baccelli, F. and Brémaud, P. (1986). Palm probabilities and queues. *Springer Lect. Notes in Stat.*

[2] Berger, A.W. (1991). Performance analysis of a rate-control throttle where tokens and jobs queue. *IEEE J. Select. Areas Comm.*, **9**, no.2, 165-170.

[3] Buttó, M., Cavallero, E. and Tonietti, A. (1991). Effectiveness of the "leaky bucket" policing mechanism in ATM networks. *IEEE J. Select. Areas Comm.*, **9**, no.3, 335-342.

[4] Dittman, L., Jacobsen, S.B. and Moth, K. (1991). Flow enforcement algorithms for ATM networks. *IEEE J. Select. Areas Comm.*, **9**, no.3, 343-350.

[5] Eckberg, A.E. and Lucantoni, D.M. (1988). A traffic/performance analysis of the bandwidth management throughput-burstiness filter. *Proc. 29th IEEE Conf. Dec. Contr.*, 2118-2123.

[6] Franken, P., König, D., Arndt, U. and Schmidt, V. (1981). *Point processes and queues.* Academie Verlag, Berlin.

[7] Rathgeb, E.P. (1991). Modeling and performance comparison of policing mechanisms for ATM networks. *IEEE J. Select. Areas Comm.*, **9**, no.3, 325-334.

ON STATIONARY QUEUEING NETWORKS WITH JOB FEEDBACK

NICHOLAS BAMBOS and KIMBERLY WASSERMAN

Department of Electrical Engineering,
University of California at Los Angeles, CA. 90024

This is a (very) brief report on some recent results concerning the stochastic behavior (and performance) of *tandem* queueing networks with stationary and ergodic service times and feedback of jobs. That is, every job after finishing service at the last queue is rerouted to the first queue, according to some feedback policy, for a random number of times before leaving the system. The asymptotic average execution time per job, with jobs requiring rerouting through the queues is computed, as their number tends to infinity. Specific conditions are identified, under which a simple feedback policy described below minimizes the average execution time per job. A dynamic scheduling policy is constructed, which stabilizes the system for the maximum possible arrival rate of jobs. The results also provide insight into the operation of general (not necessarily tandem) stationary queueing networks with feedback, whose analysis still remains a largely unsolved problem.

Tandem queueing networks with feedback of jobs provide natural models for communication systems (involving retransmission of unacknowledged or corrupted messages), computer systems (parallel computation systems, operating systems, database management systems etc. involving rerouting of functional tokens), and flexible manufacturing systems (feeding back products that fail quality control at the end of production lines, rerouting VLSI chips for multiple layer processing in VLSI manufacturing) etc. Acyclic (no-feedback) stationary queueing networks were first analyzed by Loynes [4]. Concerning non-acyclic queueing networks Koenig & Schmidt [3] have studied a single G/G/1 queue with feedback, providing one of the very few results in area. Up to now, the most powerful results on general networks extend to the case of independent service times and Markovian routing (and feedback), and have been obtained by Borovkov [1], [2] using renewal theory techniques; Sigman [5] has also obtained analogous results, using Harris-recurrence techniques.

We consider a network of $M \in \mathcal{Z}_+$ first-come-first-served queues in tandem and a random input flow of jobs arriving for processing by the servers of the queues. When a job, say the n-th one, leaves the m-th queue, where $m \in \{1,2,3,\dots M-1\}$, it immediately joins the $(m+1)$-st one. However, when it leaves the M-th one, it may be rerouted to the first queue and go through all the queues in the network again. This rerouting may happen for a random number of times, as specified below, before the job actually leaves the system.

We formulate the input to the network as a random sequence $\mathbf{N}$, which is defined on some probability space $\{\Omega, \mathcal{A}, P\}$ and is of the form

$$\mathbf{N} = \{(\tau_n, r_n, \sigma_n^{m,k}),\ n \in \mathcal{Z},\ m \in \{1,2,3,\dots M\},\ k \in \{1,2,3,\dots K\}\}, \tag{1}$$

where:
1) $\tau_n = t_{n+1} - t_n \in [0,\infty)$ is the interarrival time between the n-th and $(n+1)$-st jobs, and t_n is the arrival time of the n-th job to arrive to the first queue for the first time, that is, from outside the network (we assume that the numbering is such that $\dots t_{-1} \leq t_0 \leq t_1 = 0 \leq t_2 \dots t_n \leq t_{n+1} \dots$),
2) $r_n \in \{1,2,3,\dots K\}$, $K \in \mathcal{Z}$ is the number of times that the n-th job will go through the tandem queues, being rerouted from the last queue to the first one (the first passing is included), before it leaves the network for ever, and
3) $\sigma_n^{m,k} \in [0,\infty)$ is the service time of the n-th job in the m-th queue, during the k-th time that the n-th job qoes through the m-th queue (if the job does go through the queue that many times, and does not leave earlier). Each job can be rerouted at most $K-1$ times through the network. All the queues have infinite capacity buffers and are of the first-come-first-served (FCFS) type.

The input $\mathbf{N}$ is taken to be *stationary* and *ergodic* with respect to the sample path transformation (index shift) $\theta\mathbf{N} = \{(\tau_{n-1}, r_{n-1}, \sigma_{n-1}^{m,k}),\ n \in \mathcal{Z},\ m \in \{1,2,3,\dots M\},\ k \in \{1,2,3,\dots K\}\}$. Finally, $\lambda = 1/E[\tau]$ is the *arrival rate* of jobs to the network from the outside (not including the recycled ones).

It turns out that a basic problem that we need to study first is the following. Suppose that we gather the jobs with indices in $\mathcal{I}_{ij} = \{i, i+1, \dots j-2, j-1\}$ $(i < j)$ in the front buffer of network (that of the first queue), without any processing taking place. The jobs are placed in the buffer in increasing index order. At time 0 we start processing the jobs, following the tandem queueing discipline described before. When a job $n \in \mathcal{I}_{ij}$ completes service in the M-th queue (the last one) and, according to r_n (if $r_n > 1$), has to be rerouted through the network, it is immediately placed in the *first empty place* of the front buffer, i.e., at the back of the first queue. We denote by ϕ^* the previous *feedback policy* of rerouted jobs, and call it *First-Empty-Place* (**FEP**) policy. It is the most natural (and fair) rerouting strategy; however, there are many other possibilities, as we will discuss in the last section. For example, we could fit the rerouted job in some place between the jobs already in the buffer (of course, this would require the shifting of some jobs to adjacent places). We denote the set of all feedback policies by $\mathbf{\Phi}$. It turns out that the performance of the system depends heavily on the chosen feedback policy.

Given the above setup, define $T_{ij}^{\phi^*}$ to be the *total execution time* of the jobs with indices in $\mathcal{I}_{ij}$, given that the feedback policy is ϕ^*. Note that the first job to enter the network for the first time may not be the one to leave first, since it may have to be rerouted many times. Finally, let $\phi sup* \in \Phi$ be the feedback policy the minimizes the total execution time of jobs in $\mathcal{I}_{ij}$, and let $T_{ij}^{\phi^{opt}}$ be the corresponding minimal total execution time. We then have the following result.

Theorem: (Asymptotic Optimality of FEP Feedback Policies) For every random sequence $\mathbf{N}$, stationary and ergodic with respect to index shifts θ, such that the sequence $\{r_n,\ n \in \mathcal{Z}\}$ is independent of the sequence $\{\sigma_n^{m,k},\ n \in \mathcal{Z},\ m \in \{1,2,\ldots M\},\ k \in \{1,2,\ldots K\}\}$, and $E[\sigma^{m,k}] = E[\sigma^{m,1}]$ for every $k \in \{1,2,\ldots K\}$, $m \in \{1,2,\ldots M\}$, we have: $\lim_{j\to\infty}\left[T_{ij}^{\phi^{opt}}/(j-i)\right] = \lim_{j\to\infty}\left[T_{ij}^{\phi^*}/(j-i)\right] = E[r] \times \max_{d\in\{1,2,\ldots,M\}}\left\{E[\sigma^{d,1}]\right\} = (1/\lambda^*)$, almost surely, for every fixed $i \in \mathcal{Z}$. Therefore, the FEP policy asymptotically achieves the minimal average execution time and is as good as the optimal one.

We now turn our attention at the dynamic processing problem involving the interarrival times of the jobs, where we start processing the input $\mathbf{N}$ at time 0 (arrival time of job 1) and the system is empty before that time. We device a dynamic processing scheme that stabilizes the system almost surely for the highest possible arrival rate of jobs. It processes the jobs in groups of random size constructed as follows on a given sample path of the input: The first group G_0 consists of only one job, that is, the one arriving at $t_1 = 0$. It is sent immediately for processing in the network and leaves it at time S_1. Assuming that there are some jobs arriving in $(0, s_1]$, they form the group G_1. No element of this group starts execution before time s_1. At time s_1, that the formation of G_1 has been completed, the jobs in it start being processed according to the FEP policy; the processing of the group is completed at time s_2. The jobs arriving in $(s_1, s_2]$ (assuming there are some) form group G_2; its processing is done as above ... and so on. In general, let s_k be the time at which the formation of the G_k group is completed and the group starts being processed according to the FEP policy (no element of G_k starts execution before time s_k). The jobs in G_k are those that have arrived in $(s_{k-1}, s_k]$. Define now $\nu_k = \inf\{j \in \mathcal{Z}_+ : s_k < t_j\}$; this is the index of the first job that arrives after s_k. It is then easy to see that the processing time of G_k is $T_{\nu_{k-1}\nu_k}^{\phi^*}$. Now, if there are some jobs arriving in $(s_k, s_k + T_{\nu_{k-1}\nu_k}^{\phi^*})$, then they form group G_{k+1} and $s_{k+1} = s_k + T_{\nu_{k-1}\nu_k}^{\phi^*}$. But, what if there is no job arriving in the previous time interval? Well, in that case the G_{k+1} group contains just the first job that arrives after s_k, that is, at time t_{ν_k}. Then, $s_{k+1} = t_{\nu_k}$. Therefore, in general $s_{k+1} = \max\{s_k + T_{\nu_{k-1}\nu_k}^{\phi^*},\ t_{\nu_k}\}$, where $\nu_k = \inf\{j \in \mathcal{Z}_+ : s_k < t_j\}$. This completes the description of the processing scheme, which we denote it by $\mathcal{G}$. Let $\mathbf{F}$ be the set of all possible dynamic processing schemes. Obviously $\mathcal{G} \in \mathbf{F}$.

Now let $y_j^{\mathcal{F}}$ be the departure time of the j-th job (arriving at t_j), given that the system operates under the dynamic processing scheme $\mathcal{F} \in \mathbf{F}$. If it exists almost surely (and is constant), the limit $\lambda_d^{\mathcal{F}} = \lim_{t\to\infty}\left[\frac{1}{t}\sum_{j=1}^{\infty} I_{\{y_j^{\mathcal{F}}\in[0,t)\}}\right]$, defines the departure rate of the jobs from the network. Finally, let $N_t^{\mathcal{F}}$ be the number of jobs in the network at time $t \in (0,\ \infty)$. We then have the following result.

Theorem: For every network input N, such that the sequence $\{r_n,\ n \in \mathcal{Z}\}$ is independent of the sequence $\{(\tau_n, \sigma_n^{m,k}),\ n \in \mathcal{Z}\}$, and $E[\sigma^{m,k}] = E[\sigma^{m,1}]$ for every $k \in \{1,2,3,\ldots K\}$ and $m \in \{1,2,3,\ldots M\}$, we have:

1) If $\lambda > \lambda^*$, then $\liminf_{t\to\infty}\left[N_t^{\mathcal{F}}/t\right] \geq \lambda - \lambda^* > 0$, almost surely, for every dynamic processing scheme $\mathcal{F} \in \mathbf{F}$, and so $\lim_{t\to\infty}[N_t^{\mathcal{F}}] = +\infty$. Therefore, the system is *unstable*, no matter what processing scheme $\mathcal{F} \in \mathbf{F}$ we follow. However:

2) If $\lambda < \lambda^*$ and the system operates under $\mathcal{G}$, then $\lim_{t\to\infty}\left[N_t^{\mathcal{G}}/t\right] = 0$, almost surely, and the *departure rate* of the jobs from the network $\lambda_d^{\mathcal{G}} = \lim_{t\to\infty}\left[\frac{1}{t}\sum_{j=1}^{\infty} I_{\{y_j^{\mathcal{G}}\in[0,t)\}}\right] = \lambda$, exists and is constant almost surely, and equal to the arrival rate $\lambda = \lambda_d^{\mathcal{G}}$. The system can then be characterized as *stable*.

We conclude that the system can successfully handle input flows of arrival rate up to λ^*, without excessive backlog of jobs in the network, but for higher arrival rates it blows up.

References: [1] Borovkov, A. (1987). Limit theorems for queueing networks I. Theory of Probability and its Applications, vol. 31, no. 3, 413-427. [2] Borovkov, A. (1988). Limit theorems for queueing networks II. Theory of Probability and its Applications, vol. 32, no. 2, 257-272. [3] Koenig, D. and Schmidt, V. (1986). Limit theorems for single-server feedback queues controlled by a general class of marked point processes. Theory of Probability and its Applications, vol. 30, no. 4, 712-719. [4] Loynes, R. M. (1962). The stability of a queue with non-independent inter-arrival and service times. Proc. Cambridge Philosophical Society, 58, 497-520. [5] Sigman, Karl (1990). The stability of open queueing networks. To appear in Stochastic Processes and their Applications.

Optimal Adaptive Scheduling in Multi-class M/GI/1 Queues with Feedback

Partha P. Bhattacharya, Leonidas Georgiadis and Pantelis Tsoucas

IBM Research Division T.J. Watson Research Center Yorktown Heights, NY 10598

1 Introduction

We consider a single server queueing system as in Klimov [4]. There are M queueing nodes, each with infinite waiting room, and a server that has to be allocated to one of the nodes in a non-preemptive manner. Customers arrive to the i^{th} node according to a Poisson process of rate λ_i and after completing service at this node join the queue at node j with probability p_{ij} and depart from the system with probability $p_{i0} = 1 - \sum_{j=1}^{M} p_{ij}$. The service requirements at node i are independent and identically distributed with distribution $B_i(\cdot)$. Let Π denote the class of non-idling, non-anticipative and non-preemptive server allocation policies and $Q_i^{\pi}(t)$ denote the queue size at node i at time t when policy $\pi \in \Pi$ is followed. For $\phi_i : I\!R \mapsto I\!R$, $i \in \mathcal{M} := \{1, \cdots, M\}$, convex and continuously differentiable, consider the (pathwise) optimization problem

$$\min_{\pi \in \Pi} \limsup_{t \to \infty} \sum_{i \in \mathcal{M}} \phi_i \left(\frac{1}{t} \int_0^t Q_i^{\pi}(t) \, dt \right) \quad \text{a.s.} \tag{1}$$

Suppose now that a customer arriving at node i may belong to any one of N classes, the arrival rate of class k customers at node i is $\lambda_i^{(k)}$ and after completing service at node i, a customer of class k joins the queue at node j (as a class k customer) with probability $p_{ij}^{(k)}$ and departs with probability $p_{i0}^{(k)} = 1 - \sum_{j=1}^{M} p_{ij}^{(k)}$. The service requirements of a class k customer at node i are independent and identically distributed with (class dependent) distribution $B_i^{(k)}(\cdot)$. Let $R_k^{\pi}(n)$ denote the response time in the system of the n^{th} arriving customer from class k under a scheduling policy $\pi \in \Pi$. With $\phi_k : I\!R \mapsto I\!R$, $k \in \mathcal{N} := \{1, \cdots, N\}$, convex and continuously differentiable, consider the problem

$$\min_{\pi \in \Pi} \limsup_{n \to \infty} \sum_{k \in \mathcal{N}} \phi_k \left(\frac{1}{n} \sum_{l=1}^{n} R_k^{\pi}(l) \right) \quad \text{a.s.} \tag{2}$$

The principal aim of this paper is to provide adaptive policies that solve problems (1) and (2). The key is a certain type of conservation law (see [7]) which yields a characterization of the space of queue sizes and response times achievable by policies in Π. This region is more complicated than the base of a polymatroid, the performance space of the known work-conserving systems; however, it has enough structure (see [1]) that permits efficient solutions to deterministic separable and a class of semi-separable convex optimization problems. In this paper, we use these results together with ideas from Stochastic Approximation Theory to derive simple adaptive optimal policies. These policies use minimal knowledge of statistical parameters and are implementable with low overhead.

To achieve an equitable allocation of the server to the various classes, it is often of considerable interest to solve the following class of problems. Following the notation in the description of problem (1), for $\phi_i : I\!R \mapsto I\!R,\ i \in \mathcal{M}$ continuous and non-decreasing consider the problem

$$\min_{\pi\in\Pi}\ \max_{i\in\mathcal{M}}\ \phi_i\left(\limsup_{t\to\infty}\ \frac{1}{t}\int_0^t Q_i^{\pi}(t)\,dt\right) \quad \text{a.s.} \tag{3}$$

Similarly for the multi-class system of problem (2) and $\phi_k : I\!R \mapsto I\!R,\ k \in \mathcal{N}$ continuous and non-decreasing, consider the problem

$$\min_{\pi\in\Pi}\ \max_{k\in\mathcal{N}}\ \phi_k\left(\limsup_{n\to\infty}\ \frac{1}{n}\sum_{l=1}^{n} R_k^{\pi}(l)\right) \quad \text{a.s..} \tag{4}$$

In [1], we show that there exists certain type of "equivalences" between problems (3) and (4) and those of the type (1) and (2) respectively. This allows us to immediately deduce approximately optimal policies for (3) and (4). For the special case of a multi-class M/GI/1 queue (without feedback) we provide an optimal policy for (3).

It turns out that when $\{\phi_i\}$ are continuous and strictly increasing, this method in fact provides an approximately optimal policy for a problem that is more strict than (3). To state it define a map $\psi : I\!R^M \mapsto I\!R^M$ as

$$\psi(x) \;=\; (\phi_{i_1}(x_{i_1}), \phi_{i_2}(x_{i_2}), \cdots, \phi_{i_M}(x_{i_M}))^T$$

whenever $\phi_{i_1}(x_{i_1}) \ge \phi_{i_2}(x_{i_2}) \ge \cdots \ge \phi_{i_M}(x_{i_M})$ and denote the vector $\psi(x)$ to be lexicographically smaller than $\psi(y)$ (to be written as $\psi(x) \preceq \psi(y)$) if the following condition holds: $\psi_1(x) \le \psi_1(y)$ and $\psi_i(x) > \psi_i(y)$ for some i implies $\psi_j(x) < \psi_j(y)$ for some $j < i$. The problem then is to find a policy π^* such that for every $\pi \in \Pi$,

$$\psi\left(\limsup_{t\to\infty}\ \frac{1}{t}\int_0^t Q^{\pi^*}(t)\,dt\right) \preceq \psi\left(\limsup_{t\to\infty}\ \frac{1}{t}\int_0^t Q^{\pi}(t)\,dt\right) \quad \text{a.s..} \tag{5}$$

We provide a policy that approximately solves (5). For the special case of a multi-class $M/GI/1$ queue we provide an optimal policy for (5).

Problem (1) was considered first by Klimov [4] for linear $\{\phi_i\}$. The region of achievable queue sizes is a convex polytope (see [7]), the extreme points of which can be achieved by strict priority policies. This explains the optimality of a particular such policy as shown by Klimov. Because of the non-linearity of the cost functions, the optimal solution of (1) may belong to the interior of the polytope and can not be obtained in general by strict priority policies.

The results presented here extend earlier work [2] of the authors where problem (5) was considered for linear $\{\phi_i\}$ for a $M/GI/1$ queue. Furthermore, in [2] the policies were allowed to change scheduling decisions only at the beginning of the busy periods and the service times were assumed to have finite fourth moment. For the policies presented here we only need second moment of the service times and the decision instants can be at arbitrary stopping times of the filtration of the queue length process.

In section 2, we present a policy for problem (1) and provide a proof of its optimality. Problem (2) is treated in section 3. Approximately optimal policies for (3), (4) and (5) are presented in section 4. For lack of space, we will only outline the arguments involved in the proofs; a fuller version is under preparation.

2 An optimal policy for problem (1)

We assume that (i) the service time distributions have finite second moments, that is, $\int_0^\infty t^2 dB_i(t) < \infty$, $i \in \mathcal{M}$; (ii) with $P := (p_{ij}, 1 \le i, j \le M)$ denoting the matrix of transition probabilities, the matrix $I - P$ is invertible and (iii) with $\beta_i := \int_0^\infty t\, dB_i(t)$, $i \in \mathcal{M}$, $\beta := (\beta_1, \cdots, \beta_M)^T \in \mathbb{R}^M$ and $\lambda := (\lambda_1, \cdots, \lambda_M)^T \in \mathbb{R}^M$, the stability condition $\lambda^T(I - P)^{-1}\beta < 1$ holds.

For $S \subset \mathcal{M}$ and $i \in S$, let A_i^S denote the expected total amount of service that a job starting at node i receives before exiting nodes in S for the first time. Specifically, $A_i^{\mathcal{M}}$ denotes the service received by a job at node i during its sojourn through the network. It is easy to check from first step equations that if β^S and P^S denote the restrictions of β and P to $S \subset \mathcal{M}$, then $A^S = (I - P^S)^{-1}\beta^S$. We need some additional notation. Every $x \in \mathbb{R}^M$ can be uniquely represented by an integer $M(x)$, subsets $\{S_k(x)\}_{k=1}^{M(x)}$ and $\{S^k(x)\}_{k=1}^{M(x)}$ of $\mathcal{M}$ and reals $\{\nu_k(x)\}_{k=1}^{M(x)}$ through the following procedure: (here $\phi_i'(\cdot)$ denotes the derivative)

1. $S^1(x) := \mathcal{M}$, $k = 0$

2. $k \leftarrow k + 1$

3. Compute

$$\begin{aligned}
\nu_k(x) &:= \min \left\{ \frac{\phi_i'(x_i) - \sum_{l=1}^{k-1} \nu_l(x) A_i^{S^l(x)}}{A_i^{S^k(x)}} \; : \; i \in S^k(x) \right\} \\
S_k(x) &:= \left\{ i \in S^k(x) \; : \; \frac{\phi_i'(x_i) - \sum_{l=1}^{k-1} \nu_l(x) A_i^{S^l(x)}}{A_i^{S^k(x)}} = \nu_k(x) \right\} \\
S^{k+1}(x) &:= S^k(x) - S_k(x)
\end{aligned}$$

4. If $S^{k+1}(x) \neq \emptyset$, then go to step 2. Else, set $M(x) := k$, END

For notational convenience, we will write

$$\bar{Q}_i^\pi(t) := \frac{1}{t} \int_0^t Q_i^\pi(s) \, ds, \quad i \in \mathcal{M}; \quad \bar{Q}^\pi(t) := (\bar{Q}_1^\pi(t), \cdots, \bar{Q}_M(t)^\pi).$$

Consider the following policy π^*. Let $\{\Delta_n\}$ be an arbitrary sequence of stopping times of the process of queue lengths such that $\Delta_n \to \infty$ a.s. as $n \to \infty$. During the interval $[\Delta_n, \Delta_{n+1})$, policy π^* gives (non-preemptive) priority to $S_{i+1}(\bar{Q}^{\pi^*}(\Delta_n))$ over $S_i(\bar{Q}^{\pi^*}(\Delta_n))$, $i = 1, 2, \cdots, M(\bar{Q}^{\pi^*}(\Delta_n)) - 1$, where $M(\bar{Q}^{\pi^*}(\Delta_n))$ and the sets $\{S_k(\bar{Q}^{\pi^*}(\Delta_n))\}_{k=1}^{M(\bar{Q}^{\pi^*}(\Delta_n))}$ are determined through the procedure described above. Ties can be resolved in an arbitrary pre-determined manner. We will show that π^* solves problem (1).

We begin with a characterization of the space of queue lengths achievable by policies in Π. The following result is shown in [7].

Proposition 1 *The queue length processes $\{Q_i^\pi(\cdot)\}_{i \in \mathcal{M}}$ induced by a policy $\pi \in \Pi$ satisfy the relations*

$$\liminf_{t \to \infty} \sum_{i \in S} A_i^S \bar{Q}_i^\pi(t) \geq F(S) \;\; \forall S \subset \mathcal{M} \;\; \text{and} \;\; \lim_{t \to \infty} \sum_{i \in \mathcal{M}} A_i^S \bar{Q}_i^\pi(t) = F(\mathcal{M}) \qquad (6)$$

for some $F : 2^{\mathcal{M}} \mapsto I\!R$ (that is independent of policies in Π). An equality holds in the first relation for $S \subset \mathcal{M}$ when π gives priority to nodes in S over S^c.

Define

$$\mathcal{A}(\mathcal{M}) := \left\{ x \in I\!R^M \; : \; \sum_{i \in S} A_i^S x_i \geq F(S) \;\; \forall S \subset \mathcal{M}, \; \sum_{i \in \mathcal{M}} A_i^{\mathcal{M}} x_i = F(\mathcal{M}) \right\}.$$

Using Proposition 1, one can show that for every policy $\pi \in \Pi$,

$$\liminf_{t\to\infty} \sum_{i\in\mathcal{M}} \phi_i\left(\bar{Q}_i^{\pi}(t)\right) \geq \min_{x\in\mathcal{A}(\mathcal{M})} \sum_{i\in\mathcal{M}} \phi_i(x_i) \text{ a.s.}.$$

Let T_n be the time at which the n^{th} busy period begins in the system. From standard renewal arguments it follows that to establish the optimality of π^*, it suffices to show that

$$\limsup_{n\to\infty} \sum_{i\in\mathcal{M}} \phi_i\left(\bar{Q}_i^{\pi^*}(T_n)\right) \leq \min_{x\in\mathcal{A}(\mathcal{M})} \sum_{i\in\mathcal{M}} \phi_i(x_i) \text{ a.s.}. \tag{7}$$

This is done next.

For notational convenience, we will write

$$\theta_{ni} := \bar{Q}_i^{\pi^*}(T_n), \quad i\in\mathcal{M}; \quad \theta_n := (\theta_{n1},\cdots,\theta_{nM})^T$$

through the rest of this section. Assume that the system starts from empty at time $T_0 = 0$. With the definitions $\tau := ET_1$, $\sigma_n := T_n/(n\tau)$, $n\geq 1$, $\sigma_0 := 0$, $K_{n+1} := (T_{n+1}-T_n)/\tau$, $n \geq 1$ and

$$J_{n+1,i}(\theta_n) := \frac{1}{\tau}\int_{T_n}^{T_{n+1}} Q_i^{\pi^*}(t)dt, \; i\in\mathcal{M}, \quad J_{n+1}(\theta_n) := \left(J_{n+1,1}(\theta_n),\cdots J_{n+1,M}(\theta_n)\right), \quad n\geq 0,$$

it is easy to check that $\{\theta_n\}$ and $\{\sigma_n\}$ satisfy the relations ($\theta_0 := 0$)

$$\begin{aligned}\theta_{n+1} &= \theta_n + \frac{1}{n+1}\left[J_{n+1}(\theta_n) - \theta_n K_{n+1}\right] \\ &\quad + \frac{1}{n+1}\left[J_{n+1}(\theta_n) - \theta_n K_{n+1}\right]\left[\frac{1}{\sigma_{n+1}} - 1\right],\end{aligned} \tag{8}$$

$$\sigma_{n+1} = \sigma_n + \frac{1}{n+1}\left[K_{n+1} - \sigma_n\right], \qquad n = 0,1,\cdots. \tag{9}$$

These recursions fall within the class of Stochastic Approximation algorithms and we will use a convergence theorem for these algorithms to establish (7).

Choose $\theta^* \in \left\{x \in \mathcal{A}(\mathcal{M}) : \sum_{i\in\mathcal{M}}\phi_i(x_i) = \min_{y\in\mathcal{A}(\mathcal{M})}\sum_{i\in\mathcal{M}}\phi_i(y_i)\right\}$ and define $V : I\!R^{M+1} \mapsto I\!R$ as

$$V(\theta,\sigma) = \sum_{i\in\mathcal{M}}\left[\phi_i(\theta_i) - \phi_i(\theta_i^*)\right] + 1/2\,(1-\sigma)^2. \tag{10}$$

Since $\sigma_n \to 1$ a.s., (7) would follow if we can show that

$$\limsup_{n\to\infty} V(\theta_n,\sigma_n) \leq 0 \text{ a.s.}. \tag{11}$$

It turns out that under certain statistical assumptions, (11) holds. For reasons of conciseness, we only mention the assumptions that are most important. Define

$$f^{(1)}_{n+1}(\theta_n, \sigma_n) \;:=\; \begin{pmatrix} J_{n+1}(\theta_n) - \theta_n K_{n+1} \\ K_{n+1} - \sigma_n \end{pmatrix}$$

$$f^{(2)}_{n+1}(\theta_n, \sigma_n) \;:=\; \begin{pmatrix} [J_{n+1}(\theta_n) - \theta_n K_{n+1}] \left|\frac{1}{\sigma_{n+1}} - 1\right| \\ 0 \end{pmatrix};$$

(SA.1) There exist measurable functions $\Gamma_n : I\!R^{M+1} \mapsto I\!R^{M+1}$, $n = 1, 2, \cdots$, and an increasing sequence of bounded sets $\{B_l\}$ such that $P\left(\cup_{l=1}^{\infty} \{\Gamma_n(\theta_n, \sigma_n) \in B_l \ \forall n\}\right) = 1$. Also, for every $l \geq 1$, there is an integrable non-negative r.v. Z_l such that with $f_n := f^{(1)}_n + f^{(2)}_n$,

$$P\left(\|f_{n+1}(\theta_n, \sigma_n)\| > t \middle| \theta_n, \sigma_n\right) \leq C_l P(Z_l > t) \text{ a.s. on } \{\Gamma_n(\theta_n, \sigma_n) \in B_l\}.$$

(SA.2) V is continuously differentiable and for every $\epsilon > 0$ and $l \geq 1$,

$$\liminf_{n \to \infty} \inf_{\substack{\Gamma_n(\theta,\sigma) \in B_l \\ V(\theta,\sigma) \geq \epsilon}} E\left[\left\langle -\nabla V(\theta_n, \sigma_n),\, f^{(1)}_{n+1}(\theta_n, \sigma_n)\right\rangle \,\middle|\, \theta_n = \theta, \sigma_n = \sigma\right] > 0. \tag{12}$$

(SA.3) There exists a sequence of events $\{G_n\}$, G_{n+1} conditionally independent of $(\theta_0, \sigma_0, \cdots, \theta_n, \sigma_n)$ given (θ_n, σ_n), such that

$$\sum_{n=0}^{\infty} \frac{1}{n+1} E\left[\|f^{(2)}_{n+1}(\theta_n, \sigma_n)\| \, 1\{\Gamma_n(\theta_n, \sigma_n) \in B_l\} \, 1\{G_{n+1}\}\right] < \infty, \quad \forall\, l \geq 1.$$

In the literature, the existence of conditional second moments of $f^{(1)}_n$, [3, 6], is usually assumed instead of (SA.1); however, for this assumption to hold, services times need to have finite fourth moment. For our problem, (SA.1) can be shown to hold when the service times have finite second moment. Assumption (SA.2) and (SA.3) together state that V is a "Lyapunov function" in the sense that $V(\theta_n, \sigma_n)$ decreases in the mean as n increases.

The literature on Stochastic Approximation Algorithms contains a large number of theorems that ensure a.s. convergence of these algorithms under a variety of assumptions. However, none of them appear suitable for our problem. Conditions closest to ours appear in [5, 6] where in addition to the existence of conditional ath moment, $a > 1$, of $f^{(1)}_n$, the continuity of the conditional first moment of $f^{(1)}_n$ with respect to the space variable is required.

From the Law of iterated logarithm applied to $\{T_n - T_{n-1}\}$, we have that that for some $0 < \delta_1 < 1$, and an a.s. finite r.v. C, $|1/\sigma_n - 1| \leq C n^{-\delta_1}$, $n \geq 1$, a.s.. Using the finiteness of second moments of service times assumptions, (SA.1) and (SA.3) can be shown to hold for the choices $B_l := \left\{x \in I\!R^{M+1} : 0 \leq x_i \leq l,\ 1 \leq i \leq M+1\right\}$, $G_n := \{K_n \leq \sqrt{n}\}$,

$$\Gamma_{ni}(x) := \begin{cases} x_i & \text{if } i \leq M \\ \max\left\{x_{M+1}, n^{\delta_1}\left|1/x_{M+1} - 1\right|\right\} & \text{if } i = M+1. \end{cases}, \quad n \geq 1.$$

and $Z_l := C_1(l)\tau K_1 A_1 \ + \ C_2(l) K_1 \ + \ C_3(l), \ \ l = 1, 2, \cdots$, with $\{C_i(l)\}$ positive constants. Consider now the verification of (SA.2). We will write $E_{\theta,\sigma}$ to denote conditional expectation given $\theta_n = \theta, \sigma_n = \sigma$. It can be seen that $\bar{J}^{\pi^*}_{n+1}(\theta) \ := \ E_{\theta,\sigma}[J_{n+1}(\theta_n)]$ belongs of $\mathcal{A}(\mathcal{M})$. Consequently, using the definition of V (see (10)), we have

$$E_{\theta,\sigma}\left\langle -\nabla V(\theta_n, \sigma_n), f^{(1)}_{n+1}((\theta_n, \sigma_n))\right\rangle \ = \ -\sum_{i \in \mathcal{M}} \phi_i'(\theta_i) \left[\bar{J}^{\pi^*}_{n+1,i}(\theta) - \theta_i\right] \ + \ (1 - \sigma)^2. \quad (13)$$

For fixed $\theta \in I\!R^M$, let π^θ be the strict priority policy that gives priority to $S_{i+1}(\theta)$ over $S_i(\theta)$, $i = 1, \cdots, M(\theta) - 1$ at all times, resolving ties in the same manner as π^*. From Proposition 1 it follows that the average queue lengths $\bar{J}^{\pi^\theta}_{n+1} \ := \ E\,\frac{1}{\tau}\int_{T_1}^{T_2} Q^{\pi^\theta}(t)\, dt$, satisfy

$$\sum_{i \in S^k(\theta)} A_i^{S^k(\theta)} \bar{J}^{\pi^\theta}_{n+1,i} \ = \ F(S^k(\theta)), \quad k = 1, \cdots, M(\theta). \quad (14)$$

From the definitions of $M(\theta)$, $\{S_k(\theta)\}$, we have

$$\phi_i'(\theta_i) \ = \ \sum_{l=1}^{k} \nu_l(\theta)\, A_i^{S^l(\theta)}, \quad i \in S_k(\theta), \quad k = 1, \cdots, M(\theta) \quad (15)$$

and this implies that

$$\begin{aligned} \sum_{i \in \mathcal{M}} \phi_i'(\theta_i)\left[\bar{J}^{\pi^\theta}_{n+1,i} - \theta_i^*\right] \ &= \ \sum_{k=1}^{M(\theta)} \nu_k(\theta)\Big[\sum_{i \in S^k(\theta)} A_i^{S^k(\theta)} \bar{J}^{\pi^\theta}_{n+1,i} - \sum_{i \in S^k(\theta)} A_i^{S^k(\theta)}\, \theta_i^*\Big| \\ &= \ \sum_{k=1}^{M(\theta)} \nu_k(\theta)\Big[F(S^k(\theta)) - \sum_{i \in S^k(\theta)} A_i^{S^k(\theta)}\, \theta_i^*\Big| \ \leq \ 0; \end{aligned} \quad (16)$$

where the second equality follows from (14) and the inequality follows from the facts that $\theta^* \in \mathcal{A}(\mathcal{M})$ and $\nu_k(\theta) > 0$, $k = 2, \cdots, M(\theta)$. Now using (16) and the convexity of $\{\phi_i\}$, we have

$$\begin{aligned} -\sum_{i \in \mathcal{M}} \phi_i'(\theta_i) \left[\bar{J}^{\pi^*}_{n+1,i}(\theta) - \theta_i\right] \ &= \sum_{i \in \mathcal{M}} \phi_i'(\theta_i)(\theta_i - \theta_i^*) \ - \ \sum_{i \in \mathcal{M}} \phi_i'(\theta_i)\left[\bar{J}^{\pi^\theta}_{n+1,i} - \theta_i^*\right] \\ &\qquad - \sum_{i \in \mathcal{M}} \phi_i'(\theta_i)\left[\bar{J}^{\pi^*}_{n+1,i}(\theta) - \bar{J}^{\pi^\theta}_{n+1,i}\right] \\ &\geq \ \sum_{i \in \mathcal{M}} \left[\phi_i(\theta_i) - \phi_i(\theta_i^*)\right] \\ &\qquad - \ \sum_{i \in \mathcal{M}} \phi_i'(\theta_i)\left[\bar{J}^{\pi^*}_{n+1,i}(\theta) - \bar{J}^{\pi^\theta}_{n+1,i}\right]. \end{aligned} \quad (17)$$

From (10), (13) and (17), it follows that to establish (12), it suffices to show for $l = 1, 2, \cdots$

$$\limsup_{n\to\infty} \sup_{\substack{0\le\theta_i\le l,\ i\in\mathcal{M}\\ V(\theta,\sigma)\ge\epsilon}} \sum_{i\in\mathcal{M}} \phi_i'(\theta_i)\left[\bar{J}^{\pi^*}_{n+1,i}(\theta) - \bar{J}^{\pi^\theta}_{n+1,i}\right] = 0. \tag{18}$$

To do this, we first (re)define $\{\nu_k(x)\}$, $\{S_k(x)\}$, $\{S^k(x)\}$ exactly as before, except that in step 3, $S_k(x)$ now contains only one index, chosen arbitrarily among those indices having corresponding values equal to $\nu_k(x)$. Clearly (15) still holds and we can write (18) as

$$\limsup_{n\to\infty} \sup_{\substack{0\le\theta_i\le l,\ i\in\mathcal{M}\\ V(\theta,\sigma)\ge\epsilon}} \sum_{k=2}^{M} \nu_k(\theta) \sum_{i\in S^k(\theta)} A_i^{S^k(\theta)}\, E_{\theta,\sigma}\frac{1}{\tau}\int_{T_n}^{T_{n+1}} \left|Q_i^{\pi^*}(t) - Q_i^{\pi^\theta}(t)\right| \, dt = 0; \tag{19}$$

where we eliminated the term for $k = 1$ by virtue of the fact that $\bar{J}^{\pi^*}_{n+1}(\theta)$, $\bar{J}^{\pi^\theta}_{n+1} \in \mathcal{A}(\mathcal{M})$. Fix $\delta_1 > 0$ arbitrarily small and split the sum in (19) into $\{k \ge 2 : \nu_k(\theta) < \delta_1\}$ and $\{k \ge 2 : \nu_k(\theta) \ge \delta_1\}$. The finiteness of second moments of service times is known to imply that $\sup_{\theta\in I\!R^M} \; E_{\theta,\sigma}\int_{T_n}^{T_{n+1}} Q_i^{\pi}(t)\,dt < \infty$, $i \in \mathcal{M}$ for every $\pi \in \Pi$ and using this together with the boundedness of $\nu(\cdot)$ on B_l, it follows that it suffices to show that for every $k = 2, \cdots, M$.

$$\lim_{n\to\infty} \sup_{\theta\in H_k} E\left|\sum_{i\in S^k(\theta)} A_i^{S^k(\theta)}\frac{1}{\tau}\int_{T_n}^{T_{n+1}} \left[Q_i^{\pi^*}(t) - Q_i^{\pi^\theta}(t)\right] \, dt\right| = 0, \tag{20}$$

where $H_k := \{\theta : 0 \le \theta_i \le l,\ i \in \mathcal{M},\ \nu_k(\theta) \ge \delta_1\}$. With $N_\delta(\theta) := \{\tilde{\theta} \in I\!R^M : ||\theta - \tilde{\theta}|| < \delta\}$, we have the following result:

Lemma 1 *There exists $\delta_2 > 0$ such that $S^k(\theta) = S^k(\theta^0)$, $\theta \in N_{\delta_2}(\theta^0)$ for every $\theta^0 \in H_k$.*

From the operation of the policies π^* and π^θ, it is easy to see that if the expression inside the expectation in (20) is non-zero for some sample path, then, because of Lemma 1 , we have $||\bar{Q}^{\pi^*}(t) - \theta_n|| > \delta_2$ for some $t \in [T_n, T_{n+1})$, which implies that $\int_{T_n}^{T_{n+1}} Q_i^{\pi^*}(t)\,dt > T_n\delta_2/M$ for some $i \in \mathcal{M}$. It follows that with $B_{n+1} := T_{n+1} - T_n$ and A_{n+1} denoting the total number of arrivals during $[T_n, T_{n+1})$, the quantity in (20) is at most

$$C \lim_{n\to\infty} E\left\{B_1 A_1 1_{\{B_1A_1 > T_n\delta_2/M\}}\right\} = 0$$

for some constant C.

3 An optimal policy for problem (2)

Consider now problem (2). We first describe an equivalent formulation. Consider a (replicated) network consisting of NM nodes in which the group of nodes $D_k := \left\{(k-1)M+1, \cdots, kM\right\}$ corresponds to class k, $k \in \mathcal{N}$. Arrivals to node $(k-1)M+i$ of D_k are Poisson with rate $\lambda_i^{(k)}$ and after completing service at this node a customer joins the queue at node $(k-1)M+j$ (belonging to D_k) with probability $p_{ij}^{(k)}$ and departs from the system with probability $1 - \sum_{j\in\mathcal{M}} p_{ij}^{(k)}$. Let $\Lambda_k := \sum_{i\in\mathcal{M}} \lambda_i^{(k)}$ denote the total arrival rate of class k customers. It can be shown that it suffices to determine the policy that solves

$$\min_{\pi\in\Pi} \limsup_{t\to\infty} \sum_{k\in\mathcal{N}} \phi_k \left(\frac{1}{\Lambda_k} \frac{1}{t} \int_0^t \sum_{i\in D_k} Q_i^\pi(s)\, ds \right) \quad \text{a.s.} \tag{21}$$

For node $i \in \cup_{k\in\mathcal{N}} D_k$, let J_i be such that $i \in D_{J_i}$. Also define $\bar\phi_i(x) = \phi'_{J_i}(x)/\Lambda_{J_i}$. Represent every $x \in I\!R^{NM}$ by $\tilde M(x)$, $\{\tilde S_k(x)\}_{k=1}^{\tilde M(x)}$ and $\{\tilde S^k(x)\}_{k=1}^{\tilde M(x)}$ and $\{\tilde\nu_k(x)\}_{k=1}^{\tilde M(x)}$ as in section 2 but with $\bar\phi_i(x)$ replacing $\phi_i'(x)$. With $A_k(t)$ denoting the total number of departures from D_k by time t, set

$$\bar R_i(t) := \frac{1}{A_{J_i}(t)} \int_0^t \sum_{j\in D_{J_i}} Q_j(s)\, ds, \quad \bar R(t) := (\bar R_1(t), \cdots, \bar R_{NM}(t)). \tag{22}$$

As in section 2 it can be shown that the policy that gives priority to $\tilde S_{i+1}(\bar R(\Delta_n))$ over $\tilde S_i(\bar R(\Delta_n))$, $i = 1, \cdots, \tilde M(\bar R(\Delta_n))$ during $[\Delta_n, \Delta_{n+1})$ solves problem (2). Note that to implement the policy one needs to know the total number of customers of the various classes in the system; knowledge of the queue lengths at the individual nodes is not necessary.

4 Approximately optimal policies for problems (3)-(5)

Turning attention to problems (3) and (5), we note from Theorem 3 in [1] that an approximately optimal policy can be obtained as a solution to

$$\min_{\pi\in\Pi} \limsup_{t\to\infty} \sum_{i\in\mathcal{M}} \int_0^{\frac{1}{t}\int_{s=0}^t Q_i^\pi(s)\, ds} \phi_i^r(u)\, du \quad \text{a.s.} \tag{23}$$

for r large. But this is an instance of problem (1). Problem (4) can be handled similarly. For the special case of an $M/GI/1$ queue, corollary 3 in [1] shows that the policy that solves

$$\min_{\pi\in\Pi} \limsup_{t\to\infty} \sum_{i\in\mathcal{M}} \beta_i \int_0^{\frac{1}{t}\int_{s=0}^t Q_i^\pi(s)\, ds} \phi_i(u)\, du \quad \text{a.s.} \tag{24}$$

exactly solves (3) and (5).

A problem with the approximately optimal policies obtained by solving (23) is that the choice of r is not evident. A careful examination of the operation of the policies, however, shows that for large r they have a simple form in 'most' of the state space. Specifically, for $\delta > 0$ there exists $r_0(\delta)$ such that the optimal policies for (23) with $r \geq r_0(\delta)$, give priority to class i over class j during $[\Delta_n, \Delta_{n+1})$ if $\phi_i(\bar{Q}_i(\Delta_n)) > \phi_j(\bar{Q}_j(\Delta_n))$, whenever $\bar{Q}(\Delta_n) \in \{y \in I\!R^M : |\phi_i(y_i) - \phi_j(y_j)| > \delta,\ i \neq j\}$. This observation leads us to conjecture that the simple policy that operates as above in the whole state space is optimal for problems (3) and (5). A similar conjecture can be made for problem (4); for details see [1].

References

[1] P.P. Bhattacharya, L. Georgiadis, and P. Tsoucas. Extended polymatroids: Properties and optimization. Technical report RC 17171, IBM T.J. Watson Research Center, 1991.

[2] P.P. Bhattacharya, L. Georgiadis, P. Tsoucas, and I. Viniotis. Optimality and finite time behavior of an adaptive multi-objective scheduling algorithm. Technical report RC 15967, IBM T.J. Watson Research Center, 1990.

[3] E.G. Gladyshev. On stochastic approximation. *Theory of Probability and its Applications*, 10:275–278, 1965.

[4] G. P. Klimov. Time sharing service systems i. *Theory of Probability and Applications*, 19:532–551, 1974.

[5] H. J. Kushner and A. Shwartz. An invariant measure approach to the convergence of stochastic approximations with state dependent noise. *SIAM Journal on Control and Optimization*, 22(1):13–27, January 1984.

[6] M. Metivier. Introduction to stochastic approximation and adaptive filtering. Lecture Notes: Departimento di Mathematica, Universita di Roma, 1987.

[7] P. Tsoucas. The region of achievable performance in a model of klimov. Technical Report RC16543, IBM T. J. Watson Research Center, 1991.

Efficient Layout of a Passive, Single-Hop, Fiber-Optic Interconnection (among N stations) with Capacity $\log_2 N$

Yitzhak Birk
IBM Almaden Research Center
San Jose, California *

Abstract

A passive, single-hop, fiber-optic interconnection among n stations, each with two transmitters and one receiver, and a round-robin transmission schedule for it, which permit $\log_2 n$ concurrent, non-interfering transmissions, have recently been described. This is a substantial improvement over the previously known limit of two concurrent transmissions, but the layout of this interconnection poses a challenge both in terms of wiring complexity and power budget. We show how to lay it out using only $n \log_2 n$ fiber segments. Moreover, the ratio of transmitted to received power is n, which is optimal.

1. Introduction

1.1 Single-Hop Interconnections

A single-hop interconnection ($\mathcal{SHI}$) is a static interconnection which provides a communication path between any two stations. Such paths may be shared, but there is no routing or forwarding. Examples of such interconnections are computer buses, Ethernet, and local radio networks. The capacity of an interconnection for a given traffic pattern is the product of the transmission rate and the number of concurrent non-interfering transmissions ("*concurrency*").

Whenever each station is equipped with a single transmitter and receiver, the only possible topology is a single "bus" interconnecting all stations. With baseband transmissions, the concurrency is one. Equipping stations with multiple transmitters and receivers, however, permits the construction of a variety of $\mathcal{SHI}$'s, which fall into two classes [1]:

- *bus-oriented* $\mathcal{SHI}$'s. These can be described as a collection of buses, with each transmitter and receiver connected to exactly one bus. The sets of receivers that

---

*Presently at the Electrical Engineering Department, Technion–Israel Institute of Technology, Haifa 32000, Israel; birk@ee.technion.ac.il

can hear transmissions of any two transmitters are therefore either identical (the two transmitters are on the same bus) or disjoint (they are on different buses).

- *non-bus-oriented* $\mathcal{SHI}$'s. Here, there are at least two transmitters such that the sets of receivers that can hear them are neither identical nor disjoint.

Throughout the remainder of the paper, we will speak of m source stations, SS's, connected to n destination stations, DS's. Whenever $m = n$, however, one can think of these as n pairs of bidirectional stations.

1.2 Fiber-optic $\mathcal{SHI}$'s

A fiber-optic $\mathcal{SHI}$ consists of directional star couplers interconnected via fiber segments. A directional star coupler is a passive element with a number of inputs and a number of outputs. A signal presented at an input is split among all outputs (and only those). This is in contrast with a "star" of copper wires, in which there is no sense of direction. The directionality of the fiber-optic medium makes it most suitable for implementation of $\mathcal{SHI}$'s, particularly the non-bus-oriented ones.

In an $(x \times y)$ star coupler with equal coupling to all outputs, one would expect the power at each output, P_{out}, to be approximately P_{in}/y. Unfortunately, however,

$$\frac{P_{in}}{P_{out}} = \max\{x, y\}. \tag{1}$$

This is known as the "*fan-in*" problem in fiber optics [2].

In fiber-optic implementations with direct detection, the maximum permissible transmission rate is inversely proportional to the power loss along the path ("*path loss*") [3],[4]. (This corresponds to a requirement for a minimal amount of energy per bit at the receiver.) In view of this and the fan-in problem, it is important to take path loss into account when comparing the capacities of fiber-optic $\mathcal{SHI}$'s. For facility of exposition, we will assume lossless components, so path loss is only due to signal splitting and merging.

1.3 Bus-oriented $\mathcal{SHI}$'s

Equipping each station with c_T transmitters and c_R receivers permits the construction of up to $c_T \cdot c_R$ buses such that every bus inteconnects the same number of stations and any two stations have at least one bus in common [1]. The concurrency of such an interconnection for a uniform traffic pattern (equal traffic rate between every pair of stations) is $c_T \cdot c_R$, which is the maximum uniform-traffic concurrency of any bus-oriented $\mathcal{SHI}$ [1][5][6]. This interconnection is also optimal in terms of path loss, which is

$$\frac{P_{in}}{P_{out}} = \max\left\{\frac{n}{c_T}, \frac{m}{c_R}\right\}. \tag{2}$$

Fig. 1 depicts such an interconnection with m=n=6 and $c_T = c_R = 2$.

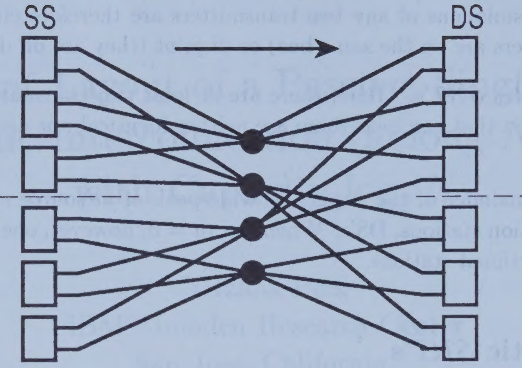


Figure 1: A single-path bus-oriented $\mathcal{SHI}$. There are $c_T \cdot c_R$ buses.

1.4 Shared Directional Multichannels

A shared directional multichannel (SDM, for short) consists of a set of inputs and a set of outputs to which we connect transmitters and receivers, respectively. A signal placed at any given input reaches a subset of the outputs. A channel is specified by the outputs reachable from each input, and it needn't be bus-oriented. (See Fig. 2.) The channel adheres to the following rules [1][7]:

1. A message transmitted into any given input of the channel reaches all outputs connected to it (and the receivers connected to those).
2. A message is received successfully by a receiver iff it is addressed to that receiver and no other signals reach it at the same time.

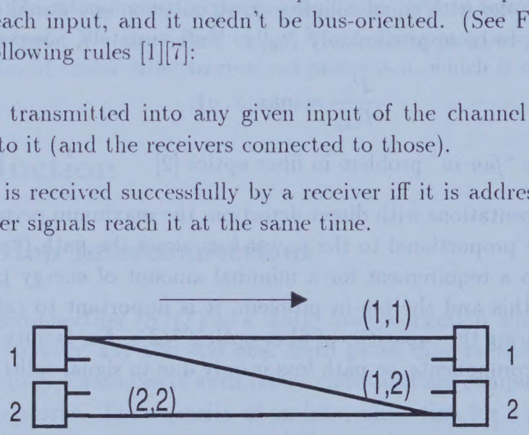


Figure 2: A Shared directional multichannel represented as a bipartite graph. Note that the subsets of receivers that can hear two transmitters are neither identical nor disjoint.

An SDM can also be used to interconnect stations with multiple transmitters and receivers. We use $(c_T, c_R; m, n)$ to denote the size of such an interconnection. For $m{=}n{=}N$, it is possible to construct an SDM whose uniform-traffic concurrency increases with N as $(\log_2 N)^{c_T+c_R-2}$ when operated with a specific round-robin transmission schedule [7]. A $(2,1;k,2^k)$ interconnection with concurrency k was described in [7]. In [6], this interconnection was augmented with another SS, resulting in a $(2,1;k{+}1,2^k)$ interconnection with concurrency $k+1$. Both can be extended to $(2,1;N,N)$ with respective uniform-traffic concurrencies $\log_2 N$ and $1+\log_2 N$.

1.5 The implementation challenge

In this paper, our primary interest is in **uniform-traffic capacity**. However, we are also interested in complexity. The measure of complexity is the number of fiber segments or star couplers. To permit a meaningful comparison, we assume that all couplers are constructed using (2×2) building blocks [8].

Since there must be some transmitter that reaches at least n/c_T receivers and some receiver that can hear at least m/c_R transmitters, a lower bound on path loss for any $(c_T, c_R; m, n)$ $\mathcal{SHI}$ is $\max\{n/c_T, m/c_R\}$. The maximum-capacity bus-oriented $\mathcal{SHI}$'s are optimal in this respect and also have minimum complexity, requiring $N \cdot c/2 \cdot log_2 N$ couplers of size (2×2) for $c_T{=}c_R{=}c$ and $m{=}n{=}N$.

The optimal layouts for bus-oriented $\mathcal{SHI}$'s occur naturally, but this is not the case for SDM-based $\mathcal{SHI}$'s whose concurrency increases with N. In fact, such an interconnection may require that each transmitter's signal be split n/c_T ways, and that m/c_R signals be combined at the input of each receiver. For the common case of $m{=}n{=}N$, this would require roughly N^2/c elementary couplers, and the path loss would be $N^2/(c_T \cdot c_R)$.

As pointed out in [6], the concurrency advantage of the general SDM-based $\mathcal{SHI}$'s over the bus-oriented ones could be offset by the need for slower transmissions. The challenge is thus to lay out a high-concurrency SDM-based $\mathcal{SHI}$ in a power-efficient manner, so that the concurrency advantage will translate into a capacity advantage. It is moreover desirable to have low complexity.

In this paper, we show how this can be done for the $(2, 1; N, N)$ SDM-based $\mathcal{SHI}$ with concurrency $\log_2 N$ and the improved one of [6]. A generalization of these results to a much broader class of interconnections will be presented elsewhere.

The remainder of the paper is organized a follows. Section 2 describes the $(2, 1; N, N)$ $\mathcal{SHI}$ originally presented in [7] as well as the modification presented in [6]. Section 3 presents an efficient layout for the interconnection of [7], section 4 shows how this can be extended to the interconnection of [6], and section 5 offers some concluding remarks.

2. Specific $(2, 1; N, N)$ Interconnections

2.1 A $(2, 1; k, 2^k)$ interconnection

Let us initially consider k SS's, each with two transmitters, connected to $n = 2^k$ DS's, each with a single receiver. We use k-bit binary vectors to identify the DS's. The k SS's are identified by the k-bit binary vectors that have a single "1".

Wiring. let $W(s, d)$ denote the transmitter ("0" or "1") used by s to reach d. Our interconnection can be described as

$$W(s, d) = s \cdot d. \tag{3}$$

In other words, the ith SS uses the ith bit in the id of each DS to decide which transmitter to connect to that DS. Fig. 3 depicts such an interconnection for $k = 4$.

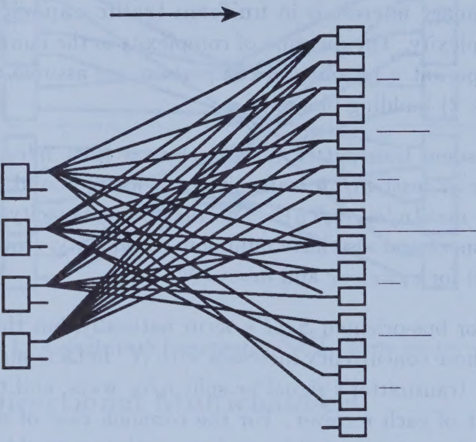


Figure 3: The logical interconnection and straightforward wiring for an interconnection of k 2-transmitter SS's and 2^k single-receiver DS's. ($k = 4$). All missing connections use 2nd-transmitters.

Schedule. $X(s,d)$ specifies the time slot in which s should transmit its message to d using the transmitter $W(s,d)$. A schedule X is *compatible* with a wiring W iff for any two pairs $(s_1,d_1) \neq (s_2,d_2)$ the following holds: if $W(s_1,d_1) = W(s_1,d_2)$, then $X(s_1,d_1) \neq X(s_2,d_2)$. One viable schedule is

$$X(s,d) = s + d. \tag{4}$$

(Modulo 2 arithmetic on the components of the k-bit vectors.) The length of this schedule is 2^k, which is indeed the number of time slots required for transmitting $k \cdot 2^k$ messages, k per slot. See [7] for generalizations and correctness proofs.

The interconnection just described can be extended to $(2,1;2^k,2^k)$ as follows. Partition the SS's into groups of k and apply the foregoing wiring function to the stations within each group. Thus, the ith stations in all groups have identical wiring. Similarly, use the same schedule function to construct a schedule for each group of SS's, and then interleave or concatenate the schedules. The attained concurrency remains k, which is equal to $log_2 N$. We will refer to SS's with identical wiring as being of the same "*type*".

A close inspection of the foregoing interconnections reveals that in any given time slot there is a "free" receiver. This is the basis of the improvement in [6]. The $(2,1;k+1,2^k)$ interconnection described there uses the same wiring function for the first k SS's. The last one chooses a transmitter based on the parity of the binary string constituting the id of the receiving station. A schedule is also presented there, but is omitted here for brevity.

3. Efficient Layout of the $(2,1;N,N)$ Interconnection

The only way to meet the implementation challenge is to overlap the splitting and merging, so that each coupler in the path has equal or nearly equal numbers of inputs and outputs. We achieve this by taking advantage of wiring symmetries among SS's, symmetries among DS's, and overlapping the splitting phase with the merging phase.

Symmetries among SS's. We may merge the outputs of individual transmitters with identical connections (there are $2k$ sets, each with N/k such transmitters) using $(\frac{N}{k} \times \frac{N}{2})$ star couplers. In so doing, we both replicate each of the signals $N/2$ times and combine sets of N/k signals that are to reach the same receivers. For any given receiver, we now take one output fiber of each of the k appropriate couplers and connect them to a $(k \times 1)$ coupler, whose output is connected to the receiver. The total path loss of this scheme is $Nk/2 = N \log_2 N/2$, a substantial improvement over the straightforward scheme. If N/k is not an integer, we add dummy SS's until their number is an integer multiple of k. In other words, we use $\lceil N/k \rceil$ for the group size.

Symmetries among DS's. Recalling that the transmitter used by the ith SS to reach DS number j is determined by the value of the ith bit in the binary representation of j, it follows that all DS's whose numbers (in binary representation) have some x bits in common have the same connections to the corresponding $(x \cdot N/k)$ SS's. The number of such DS's is 2^{k-x}. This leads us to a three-stage layout, which we now begin to construct.

Since no two DS's have identical connections, the signals must reach a DS through a coupler with one output and at least two inputs. Optimistically, we assume that stage-3 couplers are (2×1). This, in turn, implies that an output of a stage-2 coupler must carry signals from all members of some $N/2$ SS's of $k/2$ types. Specifically, we will let such a coupler carry signals from either the first or last $k/2$ types. Since the number of DS's with the first or last $k/2$ bits in common is $2^{k/2}$, this will be the number of outputs of couplers in this stage. Each 3rd-stage coupler will be connected to the outputs of the two stage-2 couplers that carry the combined signals of the transmitters to which its receiver should be connected. (An output of a coupler reached by the appropriate combination of transmitters of the first $k/2$ types of SS's and one carrying the appropriate combination from the remaining $k/2$ types.) The number of couplers in the 2nd stage will be $2 \cdot 2^{k/2}$.

Overlapping the splitting and merging. We have thus far observed that (i) the signals of up to N/k transmitters may be merged, since they all have identical destinations, (ii) a last (3rd) stage of couplers is ideally of size (2×1), and stage-2 couplers each have $2^{k/2}$ outputs. Having determined the number of outputs of couplers in stages 2 and 3, and since each transmitter must reach $2^k/2$ receivers, it follows that the number of outputs of a stage-1 coupler is $(2^{k/2-1})$.

If we took full advantage of the merging possibilities in the first stage, a stage-2 coupler would only have $k/2$ inputs, leading to suboptimal path loss. Instead, we break the merger of N/k signals into two steps: in the first step, we combine groups of x signals (transmitters with identical connections) using $x \times 2^{k/2-1}$ couplers. (x is an integer whose value has yet to be determined.) Next, we take one output of each coupler and connect those outputs to inputs of a stage-2 coupler, which (in addition to other roles) completes the merger. The number of inputs of a stage-2 coupler will therefore be $k \cdot z/2$ for some integer z. Finally, we select the best values of x and z.

Summarizing the situation:

- Stage-3 couplers are (2×1).
- stage-2 couplers have $k \cdot z/2$ inputs and $2^{k/2}$ outputs.
- stage-1 couplers have x inputs and $2^{k/2-1}$ outputs.
- $z \cdot x = N/k = 2^k/k$. (The number of SS's of a given type.)

We now need to pick an (x, z) combination that minimizes the power split. Optimistically, we begin by setting $x = 2^{k/2-1}$. This yields $z = N/(kx) = 2^k/(k \cdot 2^{k/2-1}) = 2^{k/2+1}/k$. Thus, the number of inputs of a stage-2 coupler is $(k/2) \cdot z = 2^{k/2}$. Surprisingly, this is exactly the number of outputs of stage-2 couplers, so we are able to achieve our goal. Fig. 4 depicts the optimal wiring scheme for $k = 6$. Couplers are represented by circles or ellipses, and stations by rectangles.

Final adjustments. The numbers of inputs and outputs of a coupler are all integers, yet there is no guarantee that all the expressions listed above produce integer results. We solve this problem by trying out the nearest integer values and using augmentation in the construction of the interconnection.

3.1 Examples

Example 1: $k = 6$; $N = 2^6 = 64$. (See Fig. 4.)

There are 64 stage-3 couplers, one per DS, each of size (2×1).

There are $2 \cdot 2^{k/2} = 16$ stage-2 couplers. The first eight represent all combinations of transmitter choices from the first $k/2 = 3$ "types" of SS's, and the remaining eight represent the choices from the remaining three types of SS's.

The number of outputs of a stage-2 coupler is $2^{k/2} = 8$.

The number of outputs of a stage-1 coupler is$2^{k/2-1} = 4$.

Since we want $\frac{k}{2} \cdot z = 3z = 8$, the two choices for z are 2 and 3.

Picking $z = 2$, $x = \lceil N/(k \cdot z) \rceil = 6$. The resulting power split is $6 \cdot 8 \cdot 2 = 96$.

Picking $z = 3$, $x = \lceil N/(k \cdot z) \rceil = 4$. The resulting power split is $4 \cdot 9 \cdot 2 = 72$. We therefore pick $z = 3$ and $x = 4$.

Having picked x, we augment the number of SS's to the smallest number which is an integer multiple of $k \cdot x$ and is greater than or equal to N, 72 in this case, and construct the interconnection as follows:

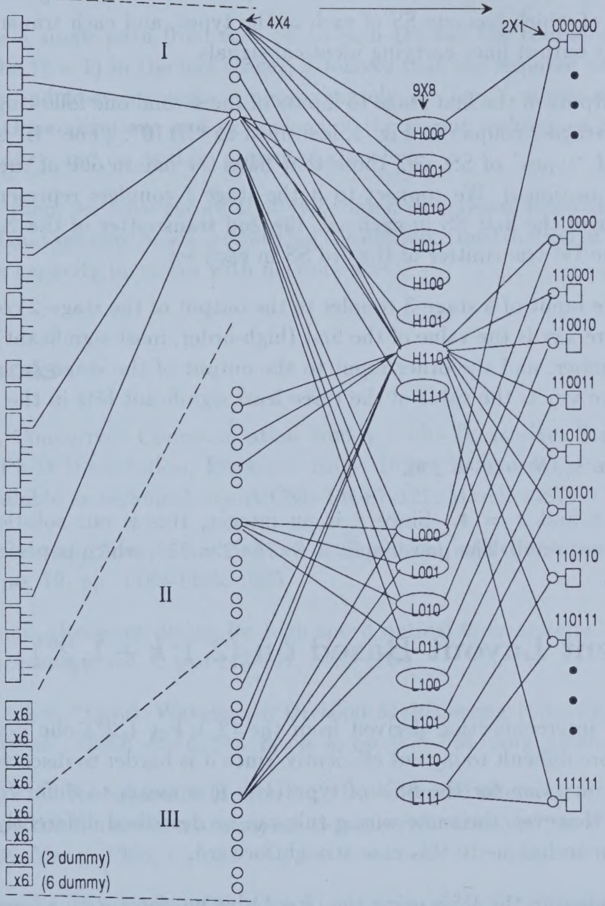

I
II
III
4X4
2X1
9X8
000000
110000
110001
110010
110011
110100
110101
110110
110111
111111
H000
H001
H010
H011
H100
H101
H110
H111
L000
L001
L010
L011
L100
L101
L110
L111
x6
x6
x6
x6
x6
x6
x6 (2 dummy)
x6 (6 dummy)

Figure 4: Wiring scheme 2 for $k = 6$; $N = 2^6 = 64$. The power split is 72.

- arrange the SS's in 12 groups of 6 (each contains one SS of each type).
- arrange the groups in 3 clusters of 4 groups.
- in each cluster, there are now 4 SS's with identical connections. Pick each set of 4 transmitters with identical connections and connect them to the four inputs of a 4×4 coupler. We can now think of the outputs of the first stage as three sets of SS's, each of which has one SS of each of the types, and each transmitter of each SS has four output lines carrying identical signals.
- connect outputs of the first stage to inputs of the second one following the example of the 7th stage-2 coupler in Fig. 4, identified as "H110". (The "H" stands for the first half of "types" of SS, i.e., those that have the one in one of the $k/2$ highest-order bit positions.) We connect to it the stage-1 couplers representing the 2nd transmitter of the first SS in each set, the 2nd transmitter of the 2nd SS in each set, and the 1st transmitter of the 3rd SS in each set.
- connect one input of a stage-3 coupler to the output of the stage-2 coupler marked Hxxx, where xxx is the value of the first (high-order, most significant) three bits in the DS number, and the other input to the output of the stage-2 coupler marked Lyyy, where yyy is the value of the three least significant bits in the DS number.

Example 2: $k = 8$; $N = 2^k = 256$.

Here, $2^{k/2-1} = 8$ and $z = 4$. Since z is an integer, this is our solution. Also, no augmentation is required. The power split is $8 \cdot 16 \cdot 2 = 256$, which is perfect.

4. Efficient Layout Based on $(2, 1; k + 1, 2^k)$

The $(2, 1; 2^k, 2^k)$ interconnection derived from the $(2, 1; k + 1, 2^k)$ one described in [6] appears to be more difficult to lay out efficiently, since it is harder to discover symmetries when the wiring function for the SS's of type $(k + 1)$ appears to differ from those for the other types. However, the same wiring rule can be described differently, making the application of our technique to this case straightforward.

We begin by numbering the DS's using the $(k + 1)$-bit numbers with an even number of "1"s. Clearly, there are exactly 2^k such numbers. Moreover, one can readily observe that the last bit of each number is equal to the parity of the string consisting of the previous bits. Finally, the number of stations which have common values in $(k + 1)/2$ of their bits is exactly half of that in an interconnection of of the original type with 2^{k+1} stations. To lay out the interconnection, we follow the same recipe as before, making the obvious modifications to accommodate the new numbers.

It is worth noting that this scheme applies most naturally to odd values of k, whereas the original one was most suitable for even ones. Nevertheless, either technique can be used in either case at some penalty.

5. Summary

We have shown how to implement a non-bus-oriented $(2,1;N,N)$ $\mathcal{SHI}$ with concurrency $\log_2 N$ and a path loss of only N, which is the absolute optimum. While we calculated path loss assuming lossless couplers, the comparison with other interconnections is equally valid for imperfect ones.

Since there is a single path from each SS to each DS and the couplers are all balanced (except for the (2×1) in the last stage), it follows that the required number of elementary couplers and fiber segments is approximately $N \cdot log_2 N$, again optimal to within a constant. Generalizations and extensions of the layout techniques presented here to other values of c_T and c_R, will be presented elsewhere.

In summary, then, we have shown that the shared directional multichannel can offer a significant advantage over buses, permitting the efficient construction of non-bus-oriented $\mathcal{SHI}$'s whose capacity increases with network size.

References

[1] Y. Birk, Concurrent Communication among Multi-Transceiver Stations via Shared Media, Ph.D Dissertation, Electrical Engr. Dept., Stanford University, Dec. 1986. Also available as technical report CSL-TR-87-321, March 1987.

[2] J.W. Goodman, "Fan-in and fan-out with optical interconnections," *Optica Acta*, vol. 32, no. 12, pp. 1489-1496, 1985.

[3] T.V. Muoi, "Receiver design for high-speed optical-fiber systems," *IEEE J. Lightwave Technology*, vol. 2, pp. 243-267, June 1984.

[4] C.A. Bracket, "Dense Wavelength Division Multiplexing Networks: Principles and Applications", *IEEE JSAC*, vol. 8, no. 6, pp. 948–964, Aug. 1990.

[5] Y. Birk, F.A. Tobagi and M.E. Marhic, "Bus-oriented interconnection topologies for single-hop communication among multi-transceiver stations", in *Proc. IEEE INFOCOM '88*, March 1988.

[6] M.T. Busche, B. Hajek, "On optical interconnection of stations having multiple transmitters and receivers" *Proc. 1990 International Symposium on Information Theory and its Applications (ISITA '90)*, Hawaii, pp. 967–970, Nov. 1990.

[7] Y. Birk, N. Linial and R. Meshulam, "On the Uniform-Traffic Capacity of Single-Hop Interconnections Employing Shared Directional Multichannels", IBM Research Report RJ 7859 (72519), Dec. '90.

[8] M.E. Marhic, *Opt. Lett.*, vol. 9, p. 368. 1984.

ON FLOW CONTROL IN OPTICAL NETWORKS

R. Bolla, F. Davoli, A. Di Febbraro

Department of Communications, Computer and Systems Science (DIST)
University of Genoa
Via Opera Pia, 11/A - 16145 Genova - Italy

Abstract. A partially reconfigurable multihop optical network is considered, with a number of traffic nodes connected by an optical medium. An asynchronous distributed algorithm is proposed to handle local partial reconfigurations with the objective of minimizing the maximum flow on the links of the network. A partial reconfiguration can be required whenever the traffic at a node exceeds a fixed threshold.

1. Introduction

The approach of the lightwave multihop architecture has been recently proposed for MANs and LANs ([1],[2],[3]) to overcome the limits imposed by the electro-optical conversions, and allow aggregate throughputs of the same order of magnitude as the enormous bandwidth made available by optical fibers (Tbits/s), through Wavelength Division Multiplexing (WDM).

A multihop network is made up by traffic nodes, each of which is provided with a number of transmitters and receivers. A link between two nodes is determined by a transmitter/receiver pair, which is assigned a wavelength, so that the links are viewed as independent channels, wavelength multiplexed on the optical medium.

The problem of partially reconfiguring this kind of networks has been recently faced by Acampora and Labourdette in [4]. Here, we propose a different model and state a control problem which consists in minimizing the maximum flow on the links by means of subsequent asynchronous partial reconfigurations.

In doing so, some links of the network are considered to be 'reallocable', as will be explained in the following. The algorithm we have designed tests the feasibility of a requested partial reconfiguration, and possibly performs it. Whenever the traffic on a link exceeds a fixed threshold, the node from which the link stems can require a partial reconfiguration. This consists in cutting an underutilized link of this node to create a new one, which connects the requesting node to the one towards which the largest portion of the traffic that exceeded the threshold was directed. The latter must be a 2-step successor of the node that has required the reconfiguration (that is, a successor of one of its successor nodes), and it will become a direct successor after the reconfiguration. Then, the algorithm sets a partial reconfiguration feasible if there is at least a wavelength that can be set free by cutting an underutilized link as described above.

The paper has the following structure. In the next section, a description of the model we propose will be given, whereas in Section 3 the optimization problem will be stated. Section 4 is devoted to the statement of the algorithm and a quite detailed explanation of the operations carried out in it. Section 5 shows that the algorithm actually performs a descent step, and Section 6 contains the conclusion.

2. Characteristics of the model

In this section the characteristics of the model of the communication network under concern will be described. First, it should be noted that, due to the nature of the problem we intend to solve, we consider networks where each node can be reached by any other traversing a finite number of intermediate nodes. In a slotted-time environment, we model a network made up by N nodes, each one of them having n_i input links and n_i output links, all with the same transmission capacity. A transmission buffer is associated with each node, in order to keep the synchronization.

This paper does not deal with the routing problems that can arise in such a network, but we suppose that these problems are handled by the so-called 'detour' algorithm [5].

We are modelling a Wavelength Optical Network (WON) in which each link corresponds to a given wavelength and to a transmitter/receiver pair. Each transmitter/receiver pair can be tuned to some different wavelengths, which means that some links can be moved. This makes the system a partially reconfigurable network.

In the network one can distinguish two different types of links. In fact, there are 'fixed' links, which guarantee that the network is completely connected in the sense mentioned above, and 'movable' links. The latter play an important role, as they can be moved from a destination node to another one, or simply removed. To move a link corresponds to change the receiver in the relevant transmitter/receiver pair, whereas if a link is removed the relevant wavelength is set free and becomes available for a possible new link. The reallocable links are associated with wavelengths close to each other, so that they occupy a 'compact' portion of the optical medium, while the rest is devoted to fixed links.

For each node i, i=1,..., N, the set of its successor nodes at time t is identified as $S(i,t)$, whereas $S_r(i,t)$ is the subset of $S(i,t)$ made up by the successor nodes of i connected to it at time t by reallocable links.

The particular nature of the network makes one think of performing some reconfiguration, with the purpose of improving link utilization and achieve fair traffic distribution, which is exactly what we do in the decision problem described in the following Section.

3. The optimization problem

The optimization problem we want to consider consists in minimizing, at every time instant t, the cost function

$$J = \max_{i=1,..,N} \{ \max_{j \in S(i,t)} \{ f_{ij}(t) \} \} \tag{1}$$

where the quantity f_{ij} represents the 'virtual' flow between nodes i and j. In fact, such a 'virtual' flow is defined as

$$f_{ij}(t) = \sum_{k \in S(j,t)} f_{ijk}^{(i)}(t) \tag{2}$$

Each quantity $f_{ijk}^{(i)}(t)$, $j \in S(i,t)$, $k \in S(j,t)$, is a 2-step 'virtual' flow measured by node i as the number of packets that had the sequence of nodes i-j-k in the primary path in their header when arriving at node i. The primary path of a packet is the path associated with it when entering the network. Whenever a packet is diverted, the alternative path associated with it by the 'detour' algorithm is kept separate from its primary path, so that it is possible to determine whether a packet is following its primary path or not.

To deal with the stated optimization problem, we try to perform an asynchronous partial reconfiguration of the network as soon as the conditions enabling it occur. In fact, whenever at a node, say node i, a flow measured on an outgoing link, say f_{ij}, grows over a fixed threshold, the node itself would be enabled to perform a partial reconfiguration, intended to make flow f_{ij} decrease, by adding a link according to a procedure that will be described in the following section.

Actually, node i will be enabled to have one more outgoing link only if it is possible to find a free wavelength to associate with it. This is the reason why, before running the procedure to add a link where and when necessary, a test on the links coming out of node i which are not involved in the request of reconfiguration is performed. If it turns out that there exists at least an outgoing link that can be cut, according to some constraints, the request of reconfiguration made by node i can be satisfied.

In the following Section the algorithm that performs both the test to find the link to be cut and the subsequent reallocation of the relevant wavelength will be stated and explained in detail.

An important feature of the network is that there is a mechanism that guarantees that only one node at a time can start a reconfiguration, which means that at every time instant only one request of reconfiguration is taken into account. This is done by fulfilling the request of the node that is the origin of the maximal flow link, among the nodes requiring a partial reconfiguration.

4. The algorithm for the partial reconfigurations

Once enabled a node to perform a partial reconfiguration of the network, we have to solve the problem of checking if there is a link going out of the node chosen that can be cut, in order to get a free wavelength to reallocate to add a new, more useful outgoing link to the same node. This test, as well as the possible subsequent cut and reallocation, is performed by the algorithm described below, where we will use some C-like statements to make it clearer.

Algorithm

0) Let i and v be two directly connected nodes for which $f_{iv} > \sigma$, where σ is a fixed threshold, and

$$k \in S(v) \qquad \text{s.t.} \qquad f_{ivk} = \max_{j \in S(v)} \{f_{ivj}\}$$

$$h \in S_r(i),\ h \neq v, \qquad \text{s.t.} \qquad f_{ih} = \min_{j \in S_r(i)} \{f_{ij}\}$$

$$h' \in S_r(i),\ h' \neq h,v, \qquad \text{s.t.} \qquad f_{ih'} = \min_{j \in S_r(i)} \{f_{ij}\}$$

$$u_m \in S(m),\ m \in S(i) \qquad \text{s.t.} \qquad f_{mu_m} = \max_{j \in S(m)} \{f_{mj}\}$$

$$\bar{f} = \max\{f_{iv} - f_{ivk};\ f_{ih} + f_{ivk};\ \max_{j \in S(i), j \neq v,h} \{f_{ij}\}\}$$

$$\bar{f}' = \max\{f_{iv} - f_{ivk};\ f_{ih'} + f_{ivk};\ \max_{j \in S(i), j \neq v,h'} \{f_{ij}\}\}$$

1) If $f_{ih} + f_{ivk} \geq f_{iv}$ then exit (no reconfiguration);

2) Let a[1..dim] be the list of node pairs $\{j,z\}$, $j \in S_r(i)$, $j \neq h,v$, $z \in S_r(j)$ s.t.
$\bar{f} + f_{ij} - f_{ijz} < f_{iv}$
ordered for increasing values of $(f_{ij} - f_{ijz})$, where dim is the number of elements in the list;

3) If Free_link_function(a[], dim) = success then:
connect node i to node k and node k to node h, and disconnect node i from node h;

```
        exit (reconfiguration performed);

4)  If $f_{ih'} + f_{ivk} \geq f_{iv}$ then exit (no reconfiguration);

5)  Let a[1..dim'] the list of node pairs {h,z}, $z \in S_r(h)$ s.t.
    $\bar{f}' + f_{ih} - f_{ihz} < f_{iv}$
    ordered for increasing values of $(f_{ih} - f_{ihz})$, where dim' is the number of elements in the list;

6)  If Free_link_function(a[], dim') = success then:
        connect node i to node k and node k to node h', and disconnect node i from node h';
        exit (reconfiguration performed);
    else exit (no reconfiguration).
```

Δ

The statement Free_link_function (a[],dim) is thought of as a call of the C-like function stated as follows

```
Free_link_function (a[], dim)
1)  Let s = 1;
2)  If $s \leq$ dim then:
        let (m,n) = a[s];
        if $f_{mu_m} + f_{mn} - f_{imn} \leq f_{iv}$ then:
            if each flow on the first alternative path from i to m is $< f_{iv}$
                disconnect node i from node m, node m from node n and connect node i to node n;
            return success;
        let s = s+1;
        go to 2);

3)  return failed.
```

Δ

It is now worth reporting some comments on the algorithm, which actually performs elementary operations. It is intended to minimize flow f_{iv}, i.e., the flow on link (i,v), which is the link that has caused the reconfiguration request.

In doing so, the idea is that of establishing a direct link between node i and node k, which is the successor node of node v towards which the greatest portion of traffic coming from i (and therefore following the path i-v-k) is sent. The wavelength to be given to the new link (i,k) can be obtained by cutting the least used movable link, say (i,h), or the 'second' least used movable link, say (i,h').

Then, in order to make node k have the same number of input and output links, and to keep node h (or h') connected to node i, a link between nodes k and h (or h') is established. The wavelength to associate with this link is obtained by performing a disallocation operation like that depicted in figures 1a and 1b. Such an operation is done only if it does not worsen the behaviour of node m, that is, the max flow at node m does not increase.

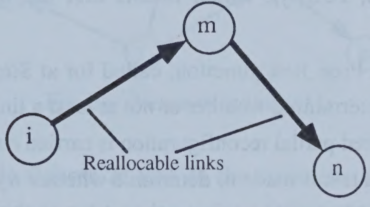


Fig. 1a. *Before disallocation*

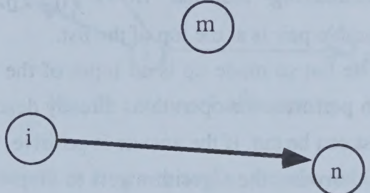


Fig. 1b. *After disallocation*

The algorithm calls for the C-like function "Free_link_function", which performs the disallocation operation described above. The input of this function is a list a[] of dimension dim, whose elements represent node pairs corresponding to nodes m, n in Fig. 1. If such a list is not empty, two tests are performed to verify whether link (m,n) and link (i,m) may be reallocated or not.

The first test checks whether or not the quantity of flow passing on (m,n) not coming from i ($f_{mn} - f_{imn}$), plus flow f_{mu_m}, which represents the maximum flow on the links coming out of node m, is less than f_{iv}. In other words, as it is not known where the traffic now passing on link (m,n) will be diverted once cut this link, it is assumed that it will be diverted on the least favourable link, then it is checked if this causes a worsening with respect to the maximum flow of the network.

To understand the meaning of the second test it must be noted that if link (i,m) is cut the flow passing on (i,m) and not directed to n, that is $f_{im} - f_{imn}$, is supposed to be diverted on the first alternative path from i to m, which becomes part of the main path in all the packets of the flow $f_{im} - f_{imn}$. If the previous control does not fail, it is then checked if the sum of each flow on that alternative path and the quantity $f_{im} - f_{imn}$ is greater than f_{iv}.

The function takes into account the elements of the list, pair after pair, until the tests described above turn out to be successful. Otherwise, the request of reconfiguration fails.

Now, we turn to explain the single steps of the algorithm. First, a test is made about the expected flow on the possible new link, once cut link (i,h). If it is greater than the present one on link (i,v) (f_{iv}), the request of reconfiguration is aborted, as it would worsen the situation at node i. Otherwise, the list of links to be given as an input to Free_link_function is made up at Step 2, by taking into account all the movable links coming out of node i, but links (i,v) and (i,h). The latter is the candidate to be cut and substituted for by link (i,k).

Note that, after the possible disallocation of (m,n), the portion of flow that presently passes on (i,m) but is not directed towards node n, which is $f_{im} - f_{imn}$, will be forced to exit node i at another output link. So as to consider the worst case, it is assumed that this flow will pass on the least advantageous link, whose flow is represented by $\bar{f}$, and node pair {j,z} is put in the list, such that the possible disallocation of links (i,j), (j,z) would result in a flow, $\bar{f} + f_{ij} - f_{ijz}$, $j \in S_r(i)$, $z \in S_r(j)$, less than the previous one, f_{iv}. Such a list is ordered according to nondecreasing 'residual' flows $f_{ij} - f_{ijz}$, $j \in S_r(i)$, $z \in S_r(j)$, which means that the most favourable pair is at the top of the list.

The list so made up is an input of the function Free_link_function, called for at Step 3, which performs the operations already described, determining whether or not at least a link in the list can be cut. If the answer is positive, the required partial reconfiguration is carried out.

Otherwise, the algorithm gets to Step 4, where a test is made to determine whether trying to deallocate link (i,h') instead of link (i,h) would lead all the same to a decrease in the maximum flow. If this is not the case, the algorithm is aborted since no reconfiguration is possible. Otherwise, at Step 5 a new list of node pairs is made up following the same criterion as in Step 2, but taking into account node h, which had been previously left out, as link (i,h) was the candidate to be cut to free a wavelength for the new link (i,k), whereas now the candidate is (i,h'). Note that only node h is now taken into account in forming the list, as all other nodes had already been considered at step 2, where $\bar{f} \le \bar{f}'$.

Finally, at Step 6 the list just made up becomes the input for function Free_link_function, which executes the same operations as in Step 3. Again, if the possibility is found to cut a link, a partial reconfiguration is carried out, otherwise the algorithm ends without affecting the virtual topology of the network.

In the next Section, it will be shown that the algorithm proposed performs a descent step of the cost function.

5. Minimization of the cost function

Suppose that at instant t_1 node i measures a flow $f_{iv}(t_1) > \sigma$ with

$$f_{iv}(t_1) = \max_{i=1,..,N} \{ \max_{j \in S(i,t_1)} \{ f_{ij}(t_1) \} \} \qquad (3)$$

and asks for a reconfiguration. Of course, the reconfiguration, if enabled, will be physically completed at instant t_2.

Our aim is now to prove that

$$\max_{i=1,..,N} \{ \max_{j\in S(i,t_2)} \{ f_{ij}(t_2)\}\} < \max_{i=1,..,N} \{ \max_{j\in S(i,t_1)} \{f_{ij}(t_1)\}\} = f_{iv}(t_1) \tag{4}$$

In doing so, first we will show that

$$\max_{j\in S(i,t_2)} \{ f_{ij}(t_2)\} < \max_{i=1,..,N} \{ \max_{j\in S(i,t_1)} \{f_{ij}(t_1)\}\} = f_{iv}(t_1) \tag{5}$$

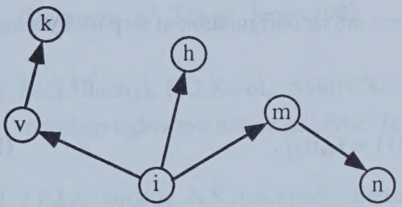


Fig. 2a. *Network at instant t_1*

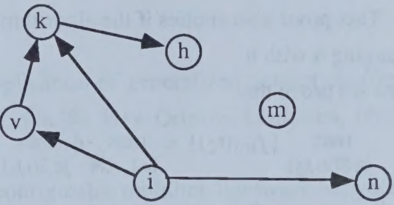


Fig. 2b. *Network at instant t_2*

In figures 2a and 2b the situations of the network at times t_1 and t_2 are shown, only considering the nodes and the links interested in the reconfiguration and supposing that the algorithm actually performs the reconfiguration at step 3). Denoting as $\hat{f}_{ij}(t_2)$, $j\in S(i,t_2)$, the flows outcoming from i at instant t_2, excluding $f_{im}(t_1) - f_{imn}(t_1)$, which has been directed on an alternative path when the link (i,m) has been cut, we can write

$$\begin{aligned}
&\hat{f}_{iv}(t_2) = f_{iv}(t_1) - f_{ivk}(t_1)\\
&\hat{f}_{ik}(t_2) = f_{ih}(t_1) + f_{ivk}(t_1)\\
&\hat{f}_{in}(t_2) = f_{imn}(t_1)\\
&\hat{f}_{ij}(t_2) = f_{ij}(t_1), \quad \forall\ j\in S(i,t_1),\ j\neq v\ ,\ h,\ m
\end{aligned} \tag{6}$$

Then $\bar{f}(t_1)$, defined in step 0) of the algorithm, becomes

$$\bar{f}(t_1) = \max\{\hat{f}_{iv}(t_2);\ \hat{f}_{ik}(t_2);\ \max_{\substack{j\in S(i,t_1)\\ j\neq v,h,m}} \{f_{ij}(t_1)\};f_{im}(t_1)\} \tag{7}$$

knowing that

$$\hat{f}_{in}(t_2) = f_{imn}(t_1) \leq f_{im}(t_1) \tag{8}$$

we obtain

$$\max_{j\in S(i,t_2)} \{\hat{f}_{ij}(t_2)\} \leq \bar{f}(t_1) \tag{9}$$

Now we consider the worst case, which is when the flow $f_{im}(t_1) - f_{imn}(t_1)$ is completely directed on the link with the maximum flow, and by using the inequality in step 2) of the algorithm, we can write

$$\max_{j\in S(i,t_2)} \{f_{ij}(t_2)\} = \max_{j\in S(i,t_2)} \{\hat{f}_{ij}(t_2)\} + f_{im}(t_1) - f_{imn}(t_1) \leq \bar{f}(t_1) + f_{im}(t_1) - f_{imn}(t_1) < f_{iv}(t_1) \quad (10)$$

which proves (5).

This proof also applies if the algorithm performs the reconfiguration at step 6), by simply changing h with h'.

Now we prove that

$$\max_{j\in S(m,t_2)} \{f_{mj}(t_2)\} < \max_{i=1,..,N} \{ \max_{j\in S(i,t_1)} \{f_{ij}(t_1)\}\} = f_{iv}(t_1) \quad (11)$$

Denoting as $\hat{f}_{mj}(t_2)$ $j\in S(i,t_2)$ the flows outcoming from m at instant t_2, excluding $f_{mn}(t_1) - f_{imn}(t_1)$, which has been diverted on an alternative path when the link (m,n) has been cut, we can write

$$\hat{f}_{mj}(t_2) = f_{mj}(t_1) \quad \forall\ j\in S(m,t_1),\ j\neq n \quad (12)$$

and then

$$\max_{j\in S(m,t_2)} \{\hat{f}_{mj}(t_2)\} = f_{mu_m} \quad (13)$$

Again considering the worst case, in which the flow $f_{im}(t_1) - f_{imn}(t_1)$ is completely directed on the link with the maximum flow, and by the first inequality in step 2) of the Free_link_function, we can write

$$\max_{j\in S(m,t_2)} \{f_{mj}(t_2)\} = f_{mu_m} + f_{im}(t_1) - f_{imn}(t_1) \leq f_{iv}(t_1) \quad (14)$$

which proves (11).

Finally, as the other links affected by the reallocation are those on the alternative path, which the second condition in step 2 of the algorithm ensures to be less than f_{iv}, and the link (k,h), for which $f_{kh}(t_2) = f_{ih}(t_1) < f_{iv}(t_1)$ by hypothesis, (4) can be considered proved.

6. Conclusions

A partially reconfigurable multihop optical network has been considered, with a number of traffic nodes connected by an optical medium. A heuristic asynchronous distributed algorithm has been proposed to handle local partial reconfigurations with the objective of minimizing the maximum flow on the links of the network. The proof that the algorithm takes a descent step of the cost function at each instance has also been given.

Simulative analysis is in progress to validate the results achieved and possibly optimize the algorithm.

References

[1] A.S.Acampora, M.J.Karol, "An overview of lightwave packet networks", *IEEE Network Magazine*, vol. 3, 1989.

[2] A.S.Acampora, "A multichannel multihop local lightwave network", *Proc. IEEE Globecom '87*, Tokyo, Japan,1987.

[3] M.G.Hluchyj, M.J.Karol, "ShuffleNet: an application of generalized perfect shuffle to multihop lightwave networks",*Proc. IEEE Infocom '88*, New Orleans, Louisiana, 1988.

[4] J.P.Labourdette, A.S.Acampora, "Partially reconfigurable multihop lightwave networks", *Proc. IEEE Globecom '90*, San Diego, CA, 1990.

[5] J.A.Bannister, M.Gerla, "Routing in large Metropolitan Area Networks based on wavelength-division multiplexing technology", *Proc. NATO Workshop on Architecture and Performance Issues of High Capacity LAN and MAN*, Sophia Antipolis, France, 1990.

On the optimality of decentralized threshold quasi-stationary routing policies in complex deterministic networks

M. Aicardi, R. Minciardi, R. Pesenti
Dipartimento di Informatica, Sistemistica e Telematica (DIST)
Via Opera Pia 11A, 16145 Genova

1. Introduction

In previous papers [1, 2, 3] concerning optimal control of simple deterministic queueing networks, the optimality of simple threshold policies (based on the emptying times of the queues) has been proved, as regards the optimization of the mean flow time over an infinite horizon. Such an optimization problem was considered with reference to a simple system consisting of two (different) parallel servers (each with its own buffer) to which incoming customer are to be routed. The customer interarrival time has been supposed to be deterministic and constant; also the service times have been assumed deterministic. It is interesting to note that threshold policies have been proved to be optimal or quasi-optimal even in connection with queueing problems which are different from the above considered [4, 5, 6].

On such a basis, one is induced to consider the application of threshold-type policies in the context of more complex queueing networks. Actually, optimal control of complex queueing networks is largely an open problem, despite significant results (see, for instance, [7]) that have been achieved for networks having a particular topology.

The difficulty of achieving optimality results has led researchers to analyze the performance of simple control policies, which are generally informationally decentralized. In this connection, for instance, Perkins and Kumar [8], and Kumar and Seidmann [9] consider the application of distributed policies in a deterministic network modelling manufacturing/assembly/disassembly systems. Their analysis concerned the stability properties of very simple policies, based on purely "local" information.

Even in this paper we are interested in developing informationally decentralized control policies in a deterministic queueing network. In this study, we will take into account the optimality results that have been obtained as regards the simple deterministic network above discussed. Actually, the main purpose of this paper is that of showing that, under certain hypotheses about the network, an optimal control policy exists which turns out to be informationally decentralized, and represented by a collection of threshold-type policies applied at the various decision nodes in the network. Besides, in this way, the determination of the optimal control policy is reduced to a purely parametric optimization problem.

The paper is organized as follows. In the next section, the basic result obtained in the previous paper is briefly recalled, and an important extension is provided, which has the purpose of allowing its use in connection with more complex networks. Then, in the third section, the considered topology is introduced and the relevant control problem is presented together with the quasi-optimality result concerning the informationally decentralized policy.

This work has been partially supported by the Italian Ministry for University and Scientific and Technological Research (MURST) within a special project on Flow Management in Production Systems.

2. Local optimal policies

This section will be devoted to a brief resume of a result already presented by the authors and to a significant extension of this result that is essential for the subsequent development of the paper.

The mentioned result refers to a system with two servers, each one with its own buffer. Each server M_i follows a first-come-first-served discipline, and, at each time instant, the system state can be represented by

$$x(t) = [w_1(t), w_2(t)]$$

where $w_i(t)$ is the emptying time of the server M_i, taking into account its present queue.

A single class of identical customers is present in the system, whose arrival process is entirely deterministic. Namely, T is the constant interarrival time of the customers. In the same way, also the service process is deterministic. Namely, S_i is the constant service time of server M_i. Moreover, we assume that the following inequalities hold:

$$\text{a)} \qquad S_2 \geq S_1 > T \tag{2.1}$$

$$\text{b)} \qquad \frac{1}{T} \leq \frac{1}{S_1} + \frac{1}{S_2} \tag{2.2}$$

Without significant loss of generality, we assume that T, S_2, S_1, are integer. Condition a) avoids triviality. Instead, condition b) can be interpreted as a "stabilizability condition" (over infinite horizon). Actually, it is patent that, for systems not fulfilling (2.2), no routing policy exists which allows an upper bound for the value of $w_1(t) + w_2(t)$.

We assume that the customers arrive and are instantaneously assigned to one of the two queues at time instants kT, k=0,1,2,.... Let $u_i(kT)$ the decision variable equal to 1 if the k-th customer is assigned (routed) to M_i, and zero otherwise. Obviously, it must be $u_1(kT) + u_2(kT) = 1$. If $u_i(kT) = 1$, then we will write $J_k \rightarrow M_i$. Then the state equation of each subsystem {waiting line plus server} is:

$$w_i(kT) = \max(w_i((k-1)T) - T, 0) + u_i(kT)S_i \quad i=1,2 \tag{2.3}$$

where $w_i(kT)$ is the emptying time of server M_i after the assignment of the customer arriving at time kT.

Let us consider the set Π_s of stationary assignment (routing) policies functions of the system state, namely policies of the type

$$u(kT) = \bar{\pi}[x(kT^-)] \qquad k = 0,1,2,\ldots \tag{2.4}$$

being $u(kT) = [u_1(kT), u_2(kT)]$, and $x(kT^-)$ the system state immediately before the assignment of the customer arrived at kT. Let us define Π_{st}, subset of Π_s, as the set of stationary stabilizing policies, i.e., policies which, independently of the initial state $x(0^-)$, do not allow $w_1(kT) + w_2(kT)$ to grow indefinitely. It is plain to understand that, under the assumed conditions about our model, the set Π_{st} is certainly not empty.

Result 2.1. [2,3] With reference to the above specified model and with respect to policies belonging to Π_{st}, in the assumption of zero initial state (i.e.,

$x(0^-) = (0,0)$), we consider the problem of determining a policy which minimizes the mean flow time of customers over infinite horizon, i.e., the cost functional

$$\bar{F} = \lim_{N\to\infty} \frac{1}{N} \sum_{k=0}^{N-1} \sum_{i=1}^{2} w_i(kT)\; u_i(kT) \tag{2.5}$$

Then, at least an optimal control policy exists having the structure

$$\bar{\pi}_1(w_1(kT^-)): \quad \text{if } w_1(kT^-) \le \theta_1 \text{ then } J_k \to M_1 \text{ else } J_k \to M_2 \tag{2.6a}$$

or the structure

$$\bar{\pi}_2(w_2(kT^-)): \quad \text{if } w_2(kT\text{-}) \le \theta_2 \text{ then } J_k \to M_2 \text{ else } J_k \to M_1 \tag{2.6b}$$

(where θ_1, θ_2 are suitable threshold values).

Δ

In this paper, we need to consider an enlargement of the class of control policies of interest. Namely, we introduce the class Π_{qs} of "quasi-stationary" policies: a quasi-stationary policy has the structure

$$u(kT) = \pi[x(kT^-), j(x(kT^-), kT) \bmod \rho(x(kT^-))] \qquad k = 0,1,2,... \tag{2.7}$$

where $j(x(kT^-),kT) - 1$ is the number of times that the state vector $x(kT^-)$ has been encountered before time instant kT^-, and $\rho(.)$ non negative mapping from **Q** to **N**. In this way, the behavior of the policy when a given state is reached is periodic. Obviously, Π_{qs} includes Π_s.

Here again, we can restrict our attention to quasi-stationary stabilizing policies. Let Π_{qst} the collection of such policies, then, the integrity assumption about the service and interarrival times implies that the application of such policies make the system cycle through a finite number of states, after a possible finite transient.

Remark 2.1. It can be deduced from the proof (reported in [3]) of the optimality Result 2.1 that the control policies (2.6) remain optimal even in the case that the set of possible policies under concern is enlarged from Π_{st} to Π_{qst}.

Δ

At this point, it is necessary to report the key result for the subsequent development of the paper.

Result 2.2. With respect to policies belonging to Π_{qst}, and in the assumption of zero initial state (i.e., $x(0^-) = (0,0)$), consider the minimization of the mean flow time of customers over infinite horizon (i.e., the cost functional (2.5)), under the additional constraint that the output rates from servers M_1 and M_2, respectively λ_1, λ_2, have a fixed ratio (obviously, it must be $\lambda_1 + \lambda_2 = \frac{1}{T}$). In connection with such a problem at least an optimal policy exists having the following structure, that we will refer to in the following as a threshold quasi-stationary policy:

$$\pi_1(w_1(kT^-)): \begin{cases} J_k \to M_1 & \text{if } w_1(kT^-) < \theta_1 \\ J_k \to M_2 & \text{if } w_1(kT^-) > \theta_1 \\ J_k \to M_i & \text{if } w_1(kT^-) = \theta_1 \end{cases} \tag{2.8}$$

where M_i is determined according to an assignment cycle which uniquely refers to assignment instants in which $w_1(kT^-) = \theta_1$ (i.e., $\rho(\alpha) = 0,\ \forall\, \alpha \neq \theta_1$). Actually such a policy can be shown to belong to Π_{qst}. Besides, a similar optimal policy exists with the server indexes interchanged.

Δ

Proof (sketch):

We will need the proof of a preliminary instrumental result that we can state as follows.

Starting from an arbitrary infinite assignment sequence representing an outcome of a policy π belonging to Π_{qst}, it is always possible to build a sequence having the following characteristics: a) it maintains the same customer assignment ratio as the original one; b) it does not worsen the customer mean flow time, as far as only customers assigned to M_1 are regarded; c) it is compatible with the application of a threshold quasi-stationary policy of structure (2.8).

The main body of this proof will be devoted to justify the above statement. That can be achieved by means of the following (conceptual) procedure:

i) decomposition of the original infinite assignment sequence into an infinite list of finite subsequences having certain characteristics;

ii) modification of any subsequence to transform it into a series of subsequences, each of which compatible with the application of a general threshold (quasi-stationary) policy (GTP)(i.e., a policy having the same structure as (2.8) but without a periodic behavior in connection with the threshold value);

iii) modification of the infinite sequence made of subsequences compatible with GTPs to obtain a new sequence where all the threshold values are the same;

iv) final modification of the sequence to make it compatible with a quasi-stationary threshold policy.

Before entering into details, it is necessary to introduce the following representation of any infinite sequence of assignments:

$$\sigma = [\{a_i, b_i\}_{i=1}^{\infty}] \qquad a_1 \geq 0,\ b_1 \geq 1; \qquad a_i \geq 1,\ b_i \geq 1 \qquad \forall i;$$

where a_i (b_i) stands for a_i (b_i) consecutive assignments to M_1 (M_2). Then following the same arguments used in [1, 2, 3] as regards stationary policies, it can be shown that:

- the application of any optimal stabilizing policy in Π_{qst} makes the state of each server assume zero value infinitely times ;
- there is no loss of optimality in restricting the class of policies of interest to policies which never assign the incoming customer to a server whose state is greater than or equal to T;

- there is no loss of optimality in restricting the class of policies of interest to policies which never assign two consecutive customers to M_2.

Then we can start the constructive proof of the above statement.

step i)

On the basis of the results above claimed, the possibility arises to partition the considered infinite sequence into an infinite series of subsequences

$$\sigma_k = [\{a_i,1\}_{i=1}^{n_k}] \qquad k=1,2,...$$

characterized by the fact that state w_1 is zero before the first assignment instant of the subsequence and always greater than zero before any other assignment instant. We will call such sequences state-zeroing (s.z.) subsequences in the following.

Then, let us consider the value

$$\theta^k = \sup_j\{w_j(kT^-) : J_j \rightarrow M_1 \text{ in } \sigma_k\}$$

obviously $\theta^k < T$;

step ii)

let us focus on a single s.z. subsequence σ_k. This can be thought as the outcome of a GTP with threshold value $\theta_1 = \theta^k$, possibly except for the last customer assignment in the subsequence. In fact, by definition of θ^k, no customer is assigned to M_1 whenever $w_1(sT^-) > \theta^k$. Moreover, by definition of σ_k, whenever $w_1(sT^-) < \theta^k < T$, this implies $J_s \rightarrow M_1$, possibly apart from the last assignment.

Then, assume that the latter possibility occurs, i.e., that a time instant sT^- exists such that $J_s \rightarrow M_1$, whereas $J_z \rightarrow M_2$, being J_z the last customer of σ_k and being $w_1(sT^-) > w_1(zT^-)$.

In this case let us represent the original s.z. subsequence as

$$\sigma_k = [\{a_i,1\}_{i=1}^{n_i} \ (a_s,1) \ \{a_j,1\}_{j=1}^{n_j} \ \{a_r,1\}_{r=1}^{n_r} \ (a_z,1)]$$

$$(n_i,n_j,n_k \geq 0, \ n_i+n_j+n_r = n_k-2)$$

where, we suppose that:

- J_s is the last customer assigned to M_1 by $(a_s,1)$;
- whenever $n_j > 0$, J_v is the last customer assigned to M_2 by $\{a_j,1\}_{j=0}^{n_j}$, and $w_1(vT^-)$ is the lowest value of the system state w_1, from sT^- onwards, for which an assignment to M_2 takes place, except for the final assignment to M_2.

The s.z. subsequence σ_k can be modified into

$$\xi_k = [\{a_i,1\}_{i=1}^{n_i} \ (a_s - 1,1) \ \{a_r,1\}_{r=1}^{n_r} \ (a_z + 1,1) \ \{a_j,1\}_{j=1}^{n_j}]$$

which has the same numbers of assignments to M_1 and to M_2 as σ_k.

ξ_k is characterized by the following facts, that can be proved by direct use of the state equation of server M_1:

- the mean flow time of customers assigned to M_1 is lowered with respect to σ_k;
- the state of server M_1 is lower than T before the last assignment in ξ_k; this ensures that the considered modification does not propagate on the system state evolution (relevant to M_1) after the end of the considered subsequence of assignments.

Note, in passing, that an idle time for server M_1 takes place after the last assignment in $(a_s - 1,1)$, and other idle times can appear in later time intervals. Thus ξ_k is not s.z. subsequences but correspond to a series of s.z. subsequences.

It is apparent that by repeated application of the above modification, we eventually transform σ_k into a finite series of s.z. subsequences each of which compatible with the application of a GTP. Of course, this transformation has to be applied to any of the subsequences of σ;

step iii)

let us consider an infinite sequence where two subsequences can be found to correspond to two different threshold values, namely

$$\sigma_r = [\{a_i,1\}_{i=1}^{n_i} \; (a_s,1)] \qquad (n_i \geq 0,\ n_i = n_r - 1)$$

$$\sigma_s = [\{a_j,1\}_{j=1}^{n_j} \; \{a_k,1\}_{k=1}^{n_k}] \qquad (n_j > 0,\ n_k \geq 0,\ n_j + n_k = n_s)$$

and let J_v the last customer assigned by σ_r, and J_z the last customer assigned to M_1 by $\{a_j,1\}_{j=1}^{n_j}$. We assume that $w_1(zT^-) > w_1(vT^-)$, where $w_1(vT^-)= \theta^r$, due to the application of step 2.

In this case, we can obtain a new sequence, not worsening the cost functional (as regards the customers routed to M_1) and maintaining the same assignment ratio. In fact, the two subsequences σ_r, σ_s can be modified into

$$\sigma'_r = [\{a_i,1\}_{i=1}^{n_i} \; (a_s+1,1) \quad \{a_k,1\}_{k=1}^{n_k}]$$

$$\sigma'_s = [\{a_j - 1,1\}_{j=1}^{n_j}]$$

which have the same numbers of assignments to M_1 and to M_2 as σ_r and σ_s. Here again, the proof that the mean flow time customers routed to M_1 does not increase can be obtained by direct use of the state equation for M_1.

Note that both σ'_r and σ'_s are s.z. subsequence,. so that there are no effects caused by the introduced substitution on the rest of the infinite sequence under concern.

By repeated application of this argument, it is possible to eventually obtain an infinite sequence made of s.z. subsequences all characterized by the same threshold value.

step iv)

let us define as $\frac{\mu_1}{\mu_2}$ the relative frequency of customer assignments to M_1 and to M_2, with regard only to assignment instants where w_1 is equal to the (unique) threshold value characterizing the infinite assignment sequence. Then, the following possibilities can take place:

- $\frac{\mu_1}{\mu_2} = 0$, $\frac{\mu_1}{\mu_2} \to \infty$; in this case we can substitute the original sequence with a new sequence having the same mean flow time (for customers routed to M_1) and compatible with an assignment policy having constant behavior in coincidence with the occurrence of the threshold value;
- $0 < \frac{\mu_1}{\mu_2} < \infty$;; in this case by following a similar argument as in step iii), one can show that, by mutual exchange of positions between successive assignments made when the state is equal to the threshold value, it is possible to maintain the ratio $\frac{\lambda_1}{\lambda_2}$ of the original sequence while imposing a periodic behavior of the policy in the considered "critical" state condition.

At this point we have proved the preliminary result above stated. By following a similar reasoning line, a dual procedure can be proved to exist, as regards the optimization of the mean flow time of customers routed to M_2, and the application of quasi-stationary threshold policies, having the same structure as (2.8), but referred to the state of server M_2.

Then, it could be easily shown that the application to a given sequence of the various modification steps above listed leaves a sequence compatible with the application of a quasi-stationary threshold policy over the state of M_1 as well as the application of a quasi-stationary threshold policy over the state of M_2. The same could be shown as regards the application of the dual procedure above mentioned.

From such considerations, it appears that there is no loss of optimality in considering only quasi-stationary threshold policy like (2.8), and then the result is proved.

Δ

It is interesting to note that the application of such optimal policies yields assignment sequences that are the same as those that were proved to be optimal by Hajek [10] in connection with a different optimization problem.

On the basis of Result 3.2, we can state a further result which generalizes Result 2.1.

Δ

Result 2.3. Consider a model which differs from the one to which Result 2.1 refers uniquely because of the assumption of an additional delay affecting customers leaving servers M_1, M_2. The delay affecting customers leaving M_i, say D_i, is supposed to be related only to average output flow (λ_i) from M_i. Thus the customer mean flow time is

$$F = \lim_{N\to\infty} \frac{1}{N} \sum_{k=0}^{N-1} \sum_{i=1}^{2} (w_i(kT) + D_i(\lambda_i))\, u_i(kT) \tag{2.9}$$

In the assumption of zero initial state (i.e., $x(0^-) = (0,0)$), we consider the problem of determining a policy belonging to Π_{qst} which minimizes (2.9). Then, at least an optimal policies exists having the structure (2.8).

Δ

Proof (sketch). Consider an optimal policy $\pi°$, and let $\lambda_1(\pi°)$ and $\lambda_2(\pi°)$ the output rates associated to $\pi°$. Consider now the problem defined in connection with Result 2.2, in the assumption that the values to which the output rates are fixed are just $\lambda_1(\pi°)$ and $\lambda_2(\pi°)$. Then, either $\pi°$ has the structure (2.8), or not. In the latter case Result 2.2 guarantees the existence of a policy π' having the structure (2.8) and corresponding to the same value of the cost functional as the optimal policy $\pi°$.

Δ

3. Application of threshold quasi-stationary routing policies in a class of complex networks

In the previous section a class of quasi-stationary routing policies has been introduced and proved to be optimal in the very simple network considered. In this section, we want to discuss the application of such policies to more complex networks.

The class of networks that we consider here is that of open queueing networks which can be thought of as composed of interconnected "isles," where isle consists of:

a) a decision (routing) node, where incoming customers are routed to one among (two or more) downstream parallel queues;

b) a number of parallel downstream servers each one with its own buffer.

The various isles are interconnected by means of a service/transportation system whose function is that of providing both external and internal flow to the isles. Our basic assumption is:

> one of the functions of the transportation system, which has an unlimited capacity, is that of producing an overall input flow to each isle characterized by constant interarrival times (of course, such interarrival times may be different from an isle to another).

It is also assumed that the customer routes are a-priori completely free and are determined only by on line routing decision processes.

The results provided in previous papers and that in the previous section refer to optimization problems in simple networks with only two downstream parallel servers. For this reason we will suppose here that all the isles are characterized by such a topology. In addition, we assume that our network is acyclic. A simple example of a network complying with the considered assumptions is that depicted in fig. 1. Note that the unique possibilities of branching in the network are located within the isles.

Our objective is to minimize the customer mean flow time over the entire network and over an infinite horizon. Obviously, the time spent by a customer in the whole network may be subdivided into:

a) time spent in the isles visited;

b) time spent in the transportation system.

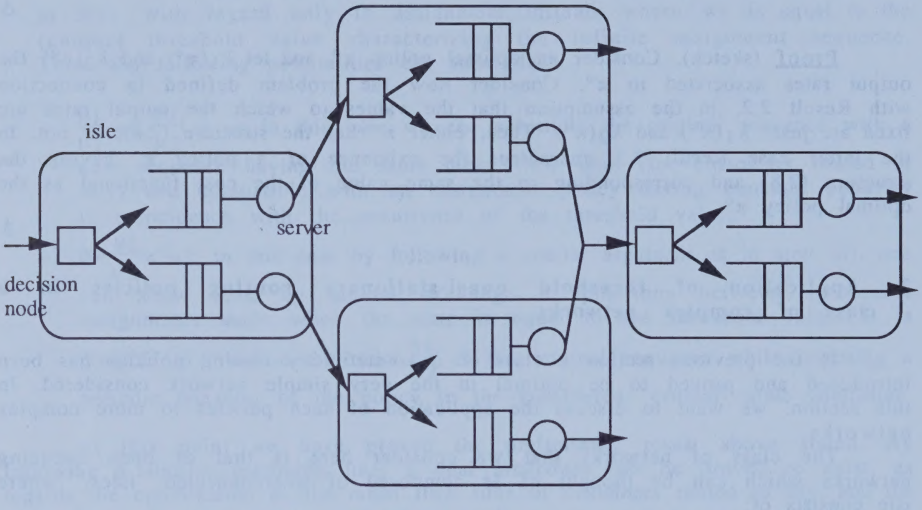


Figure 1: example of considered network.

Even within the above assumptions the statement and the solution of the mentioned optimization problem appears to be a very hard one. For this reason, we restrict ourselves to the following optimization problem.

Problem 3.1. Within the above assumptions about the system structure and the characteristics of the transportation system, minimize the mean time of type a) spent by a customer in the system, over an infinite horizon, with respect to centralized quasi-stationary routing policies.

Δ

In the statement of the above problem, the concept of quasi-stationary centralized routing policy for the overall network is used. such a concept can be immediately derived as a generalization of (2.7). Obviously, in the centralized case, the knowledge of the system state regards the emptying times in the whole network, and the routing decisions are relevant to any of the nodes.

Then, we can state the following result

Result 3.1. With respect to the optimization Problem 3.1, at least an optimal policy exists which turns out to be informationally decentralized, and consists, for each isle, of a threshold quasi-stationary policy as that in Result 3.2.

Δ

Proof (sketch). An optimal centralized (quasi-stationary) routing policy fixes the input rate of each isle, as well as the output rate of each queue. Then following the same reasoning line as in the proof of Result 2.3, we consider for each isle an optimization problem in which the input and output rates are fixed to the values corresponding to the application of the optimal (centralized) policy. As a consequence, on the basis of Result 2.2, we can conclude that the considered centralized policy can be substituted by a decentralized policy having the claimed structure and yielding the same value of the cost.

Δ

4. Conclusions

In this paper we have provided some preliminary results concerning optimal control of complex queueing networks with deterministic service times. The key result of the paper consists of a generalization of an optimality result which was provided by the authors in previous papers, and concerned the optimization of the customer mean flow time, with respect to stationary routing policies over an infinite horizon, in very simple deterministic networks.

Then, complex networks having a particular type of topology have been introduced, within the basic assumption that the transportation system makes the customer interarrival time constant for any decision node. We have considered the objective of minimizing the mean time spent by the customers in being served or waiting for service. For such a problem, we have proved that an optimal routing policy exists which turns out to be decentralized and represented by a collection of threshold quasi-stationary policies, at the various decision nodes.

References

[1] M. Aicardi, R. Minciardi, R. Pesenti, "Optimal routing in a simple network with deterministic arrivals and service times," Proc. 28th Allerton Conference on Communications, Control and Computing, Urbana Champaign,Il (USA), October 1990, pp. 640-649.

[2] M. Aicardi, R. Minciardi, R. Pesenti, "Infinite horizon optimal routing policies for simple deterministic networks," to be presented at IFAC Symposium on Design Methods of Control Systems, Zurich, Switzerland, September 1991.

[3] M. Aicardi, R. Minciardi, R. Pesenti, "Optimality of threshold policies in a simple deterministic network," Proc. European Control Conference, Grenoble, July 1991.

[4] W. Lin, P.R. Kumar, "Optimal control of queueing system with two heterogeneous servers," IEEE Trans. Autom. Contr., vol. 29, pp.696-703, 1984.

[5] J. Walrand, "A note on 'Optimal control of queueing system with two heterogeneous servers,'" Syst. & Contr. Lett., vol.4, pp.133-134, 1984.

[6] I. Viniotis, A. Ephremides, "Extension of the optimality of the threshold policy in heterogeneous multiserver queueing systems," IEEE Trans. on Autom. Contr., vol. 33, pp.104-109, 1988.

[7] B. Hajek, "Optimal Control of two interacting service stations," IEEE Trans. on Autom. Control 29, 491-499, 1985.

[8] J. Perkins, P.R. Kumar, "Stable, Distributed, Real-Time Scheduling of Flexible Manufacturing/Assembly/Disassembly Systems," IEEE Trans. on Autom. Control 34, 139-148, 1989.

[9] P.R. Kumar, T.I. Seidman, "Dynamic Instabilities and Stabilization Methods in Distributed Real-time Scheduling of Manufacturing Systems," IEEE Trans. on Autom. Control 35, 289-298, 1990.

[10] B. Hajek, "Extremal Splitting of Point Processes," Math. Oper. Res. 10, 543-556, 1985.

A DS/CDMA PACKET ERROR PROBABILITY APPROXIMATION

Floyd Simpson and Jack Holtzman
Wireless Information Network Laboratory (WINLAB)
Department of Electrical and Computer Engineering
Rutgers, The State University of New Jersey
P.O. Box 909, Piscataway, NJ 08855-0909

Abstract

The calculation of the bit or packet error probability for direct sequence (DS) code division multiple access or CDMA has in the past been analytically difficult and computationally intensive. Therefore, the results in this area generally fall into two categories: error bounds or error approximations. In this work, DS/CDMA packet error probability is considered. For a packet with block error control, we derive a packet error probability approximation that is simple to compute and is quite accurate.

I. Introduction

The calculation of the bit or packet error probabilities for DS/CDMA has in the past been analytically difficult and computationally intensive. Nevertheless, research in this area has been steady and fruitful. One of the early approximations to the bit error probability was the so called "standard Gaussian approximation" wherein the multiple access interference (MAI) due to other users was treated as an additive white Gaussian noise. Then, the probability of error is easily computed as a function of signal to noise ratio. See [1]. Other works on bounds and approximations include [2, 3, 4, 5, 6].

In [4], an "improved Gaussian approximation" was derived for the bit error probability. This approximation, as the name implies, results from further analysis of the shortcomings of the standard Gaussian approximation. This shortcoming is the central limit theorem on which the standard Gaussian approximation is based. Reference [4] showed that the Gaussian assumption for the MAI is not always accurate. Using this fact, the authors proceeded to develop the aforementioned improved Gaussian approximation to the bit error probability. They also noted that it allows for handling the bit-to-bit error dependence within packets. In [7], the results of [4] were used to derive a computationally simpler and accurate approximation to the bit error probability.

In this work, the results of [4] for approximating the probability of packet error (PPE) or its complement, the probability of packet success (PPS), are used to derive a computationally simpler and accurate approximation to the PPS. This approximation is another application of the one in [7]. There, it was stated that only key moments need to be considered, i.e., no distributional information is needed *provided* certain prior distributional analysis is done. Then, first and second moments are sufficient. The prior distributional analysis that we use has already been done in references [3, 4].

The organization of this paper is as follows: Section II describes the system model. Section III discusses the probability of packet success. In section IV, a computationally simpler and accurate approximation is presented. In section V, comparisons are drawn and illustrated via plots of probability of packet success vs. number of simultaneous users. We also examine the accuracy of our approximation, and when it can be used. Concluding remarks are given in section VI.

II. System Model

The CDMA system model considered is given in [4], and the model in [4] in turn is similar to the one used in [3] and [1]. If there are K users in the system, the kth user's transmitted signal is of the form

$$s_k(t-\tau_k) = \sqrt{2P}\,b_k(t-\tau_k)a_k(t-\tau_k)\cos(\omega_c t + \phi_k) \tag{1}$$

where $b_k(t)$ and $a_k(t)$ are the data and spreading signals, respectively, P is the received signal power, and ω_c is the carrier frequency, τ_k is the relative time delay, and ω_k is the relative phase of the carrier. The data signal $b_k(t)$ is a sequence of unit amplitude, positive and negative rectangular pulses. The spreading signal $a_k(t)$ is a faster sequence ("signature sequence") of unit amplitude, positive and negative rectangular pulses of duration T_c, called *chips*. There are N chips per data pulse. The data pulse and the chip amplitudes are all *iid* random variables with probability of 1/2 of being ± 1. During demodulation, the composite signal from all K users is multiplied by a synchronized replica of the original signature sequence, thus canceling the sequence. What results is the desired data.

The decision statistic for user 1 (the *desired receiver*), normalized with respect to the chip duration $T_c = 1$, and with all signal's received power fixed at $P = 2$, is [3, 4]:

$$Z_1 = N + \sum_{k=2}^{K} W_k \cos\Phi_k \tag{2}$$

where W_k is given by

$$W_k = P_k S_k + Q_k(1-S_k) + X_k + Y_k(1-2S_k) \tag{3}$$

with $0 \le S_k < T_c = 1$. Relative to the desired signal, the kth interfering user has a time offset to the nearest chip given by S_k and carrier phase Φ_k. The random variables P_k and Q_k are uniform on $\{-1,1\}$, and X_k and Y_k have densities

$$\begin{aligned} p_{X_k}(i) &= \binom{A}{\frac{i+A}{2}} 2^{-A} \\ & i \in \{-A, -A+2, \ldots, A-2, A\} \end{aligned} \tag{4}$$

and

$$p_{Y_k}(j) = \binom{B}{\frac{j+B}{2}} 2^{-B} \tag{5}$$

$$j \in \{-B, -B+2, \ldots, B-2, B\}.$$

A and B are related to $C = C_{1,1}(1)$, the discrete aperiodic autocorrelation of the signature sequence of the desired receiver, offset by one chip by

$$A = \frac{N-1+C}{2} \tag{6}$$

and

$$B = \frac{N-1-C}{2} \tag{7}$$

Letting $a_j^{(k)}$ be the jth chip in the signature sequence of signal k, we have that

$$C = \sum_{j=0}^{N-2} a_j^{(1)} a_{j+1}^{(1)} \tag{8}$$

with density (for random desired sequence)

$$p_C(i) \;=\; \binom{N}{\frac{i+N-1}{2}} 2^{1-N} \qquad i \in \{1-N, 3-N, \ldots, N-3, N-1\}. \tag{9}$$

As noted in [4], the random variables P_k, Q_k, X_k, and Y_k that make up W_k are conditionally independent given C. Also, a signature sequence of length N has $N-1$ chip boundaries at which the sequence may or may not change to a different value. The parameter B represents the number of chip boundaries at which a transition to a different value occurs. Therefore, B can be regarded as a measure of the amount of "spreading" given to the desired signal.

III. Probability of Packet Success

If a packet of length L bits is transmitted over a memoryless binary symmetric communication channel with average probability of data bit success Q_e, and if the packet includes block error control capability that can correct t or fewer errors, then, assuming independence, the probability of packet success Q_E becomes

$$Q_E = \sum_{i=0}^{t} \binom{L}{i} (1-Q_e)^i (Q_e)^{L-i} \tag{10}$$

If bit-to-bit error dependencies exist in the packet, then (10) cannot be used directly. In [4], an approximation that accounts for dependencies that occur if the relative delays S_k and phases Φ_k of the interfering signals are constant for the duration of a packet is developed.

The approximation used results from the development of the "improved Gaussian approximation" for bit error probability. From equations (2) and (3), and the fact that if conditioning is done on S_k (the interfering signal delay), Φ_k (phase), and on B, then the MAI becomes independent from bit-to-bit within a packet. This allows for finding the conditional PPS. Then, by unconditioning, Q_E can be easily found.

Specifically, let Φ_k and S_k be random at the start of a desired packet, but remain constant for the duration of the packet, and let B remain fixed during a desired packet. Also, let $S = (S_2, \ldots, S_K)$ and $\Phi = (\Phi_2, \ldots, \Phi_K)$ be the delays and phases of all the interfering signals. Then, each outcome $\Psi = \psi$ of the conditional MAI variance is produced by specific outcomes of S,Φ, and B. Hence, $\Phi(N/\sqrt{\psi})$ is an accurate approximation to the data bit error and $g(\Phi(N/\sqrt{\psi}); L, t)$ is an accurate approximation to the conditional PPS, where $g(x; L, t)$ = RHS of (10) with x replacing Q_e, and $\Phi(x) = 1 - Q(x)$. Unconditioning $g(\Phi(N/\sqrt{\psi}); L, t)$ yields the unconditional PPS as

$$\begin{aligned} Q_E &= \mathrm{E}_\Psi\left[g\left(\Phi\left[\frac{N}{\sqrt{\Psi}}\right]; L, t\right)\right] \\ &= \int_0^\infty g\left(\Phi\left[\frac{N}{\sqrt{\psi}}\right]; L, t\right) f_\Psi(\psi)\, d\psi \end{aligned} \tag{11}$$

where $f_\Psi(\psi)$ is the density of the total MAI variance, and E_Ψ denotes mathematical expectation wrt Ψ. See [4] for further details of assumptions used and specifics of the "improved Gaussian approximation".

The expression given in (11) generally has to be computed using numerical integration, and even for modest values of K and N, the computation time is considerable. Therefore, much simpler, and, hopefully, accurate approximations are sought. Reference [4] presents three other simpler but less accurate approximations. In the next section, we present another approximation that is very simple and quite accurate.

IV. An Accurate and Simpler Approximation

A simpler and more accurate approximation results from noting that equation (11) is the expectation of a function of a random variable (Ψ, the total MAI variance). A method was used in [7] that yields accurate approximations to expectations of functions of random variables. The approximation for $\mathrm{E}[F(\theta)]$ used in [7] is:

$$\mathrm{E}[F(\theta)] \approx \frac{2}{3}F(\mu) + \frac{1}{6}F(\mu + \sqrt{3}\sigma) + \frac{1}{6}F(\mu - \sqrt{3}\sigma) \tag{12}$$

where $F(\theta)$ is a real function of the random variable θ, μ is the mean of X, and σ is the standard deviation.

To apply the above approximation to (11), we need $\mu = \mathrm{E}[\Psi]$ and $\sigma^2 = \mathrm{VAR}[\Psi]$. From [7], μ and σ^2 are given by:

$$\begin{aligned} \mu &= (K-1)\frac{N}{3} && (13) \\ \sigma^2 &= (K-1)\left[\frac{23}{360}N^2 + N\left(\frac{1}{20} + \frac{K-2}{36}\right) - \frac{1}{20} - \frac{K-2}{36}\right] && (14) \end{aligned}$$

Then, by (12), an accurate approximation to Q_E becomes

$$\hat{Q}_E = \frac{2}{3}g\left(\Phi\left[\frac{N}{\sqrt{\mu}}\right]; L, t\right) + \frac{1}{6}g\left(\Phi\left[\frac{N}{\sqrt{\mu - \sqrt{3}\sigma}}\right]; L, t\right) + \frac{1}{6}g\left(\Phi\left[\frac{N}{\sqrt{\mu + \sqrt{3}\sigma}}\right]; L, t\right) \tag{15}$$

V. Comparisons

The approximation developed in [4] is plotted on the same plot with the approximation developed in section IV for $t = 0$, and $t = 10$. Figure 1 shows the plot of PPS (probability of packet success) vs. K (number of users) for $t = 0$. Figure 2 shows the plot for $t = 10$.

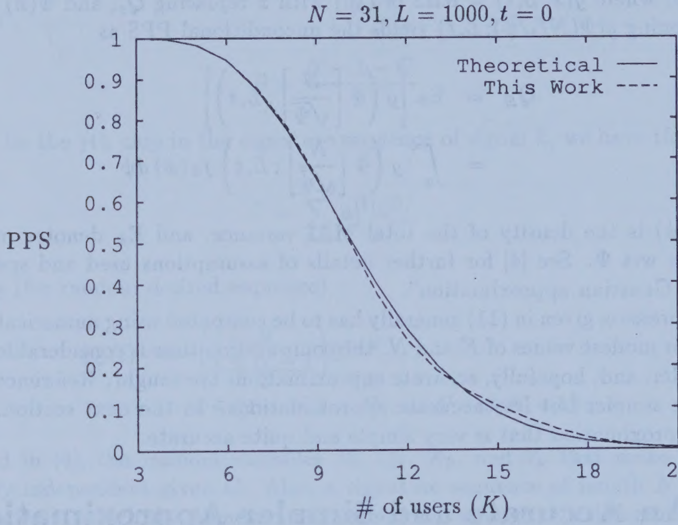


Figure 1: Q_E *and* $\hat{Q}_E$

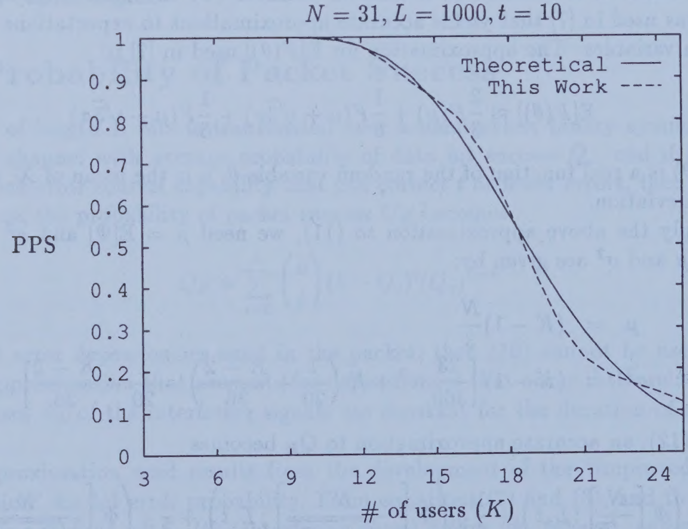


Figure 2: Q_E *and* $\hat{Q}_E$

To compare our approximation to other approximations, we have plotted the other three approximations of [4]. Figure 3 shows the results for $t = 0$, and figure 4 shows the results for $t = 10$. From figures 3 and 4, it can be seen that our approximation outperforms approximations 1–3.

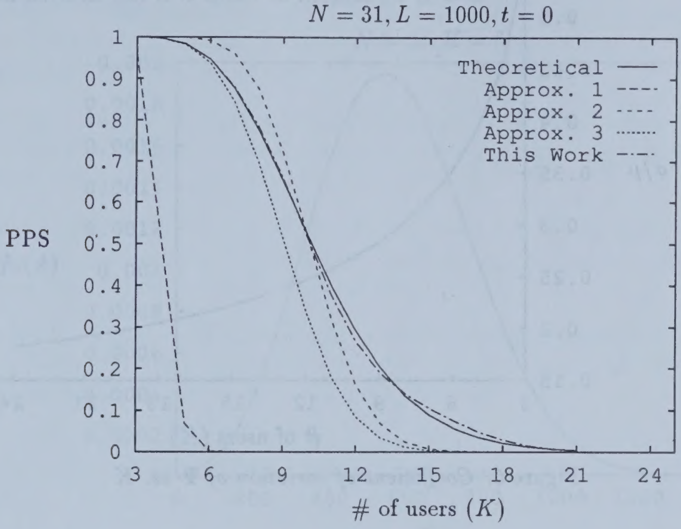


Figure 3: *All approximations*

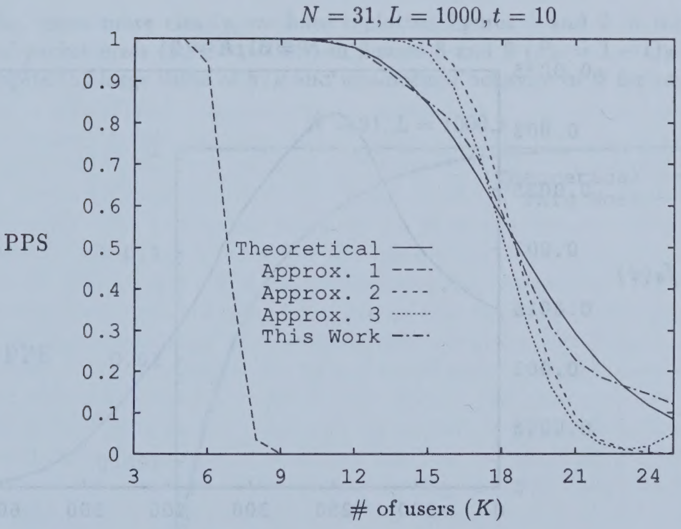


Figure 4: *All approximations*

The result in (12) is exact for fifth degree polynomials and normally distributed random variables. It is quite accurate for some (but not necessarily all) deviations from those assumptions. For the present application, the use of (12) might be suspect as σ/μ increases to $1/\sqrt{3} = 0.577$ in which case the argument $(\mu - \sqrt{3}\sigma)$ in (15) goes to zero. σ/μ is plotted in figure 5.

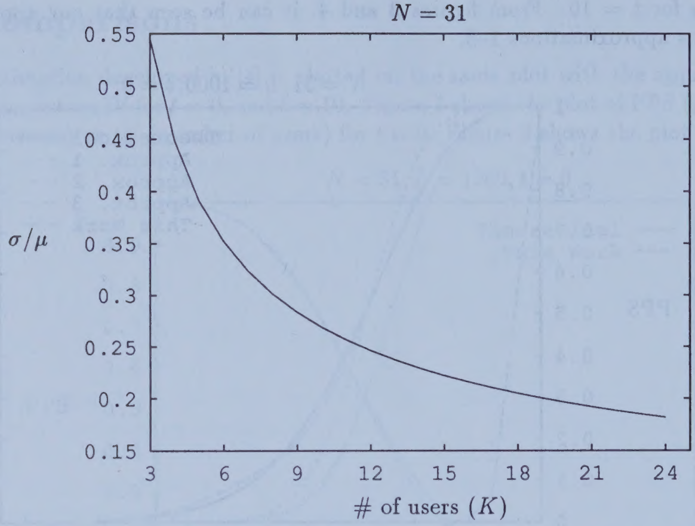


Figure 5: *Coefficient of variation of Ψ vs. K*

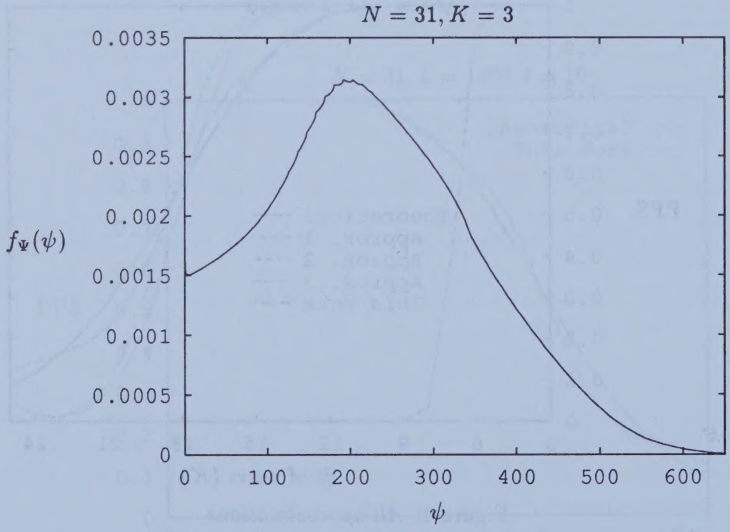


Figure 6: *Density fcn. of Ψ for $K = 3$*

It is seen that σ/μ is quite large compared to 0.577 for small K. Figures 6 and 7 shows two examples of the density function of Ψ. For $K = 3$, the density function is distinctly non-normal, but it is closer to normal for $K = 6$.

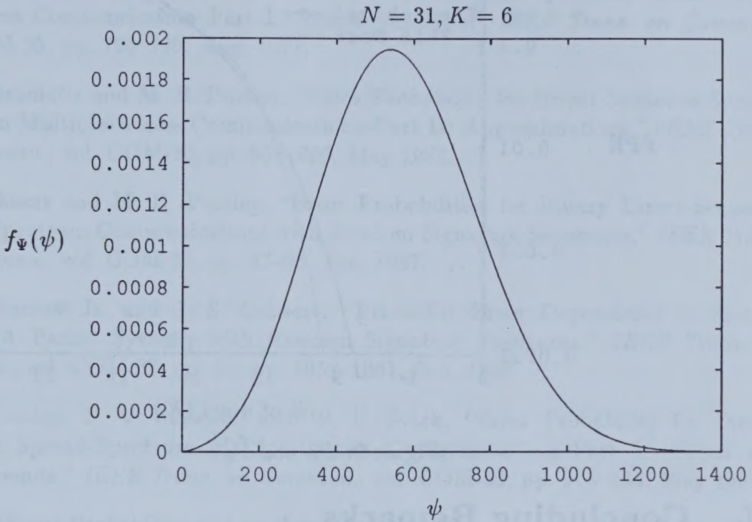


Figure 7: *Density fcn. of* Ψ *for* $K = 6$

To see the errors more clearly, we have replotted figures 1 and 2 in terms of the probability of packet error (PPE=1−PPS) in figures 8 and 9 ($P_E = 1 - Q_E$ and $\hat{P}_E = 1 - \hat{Q}_E$). Despite the large value of σ/μ and non-normal behavior of Ψ for small K, the results are quite accurate, except for certain low values of PPE.

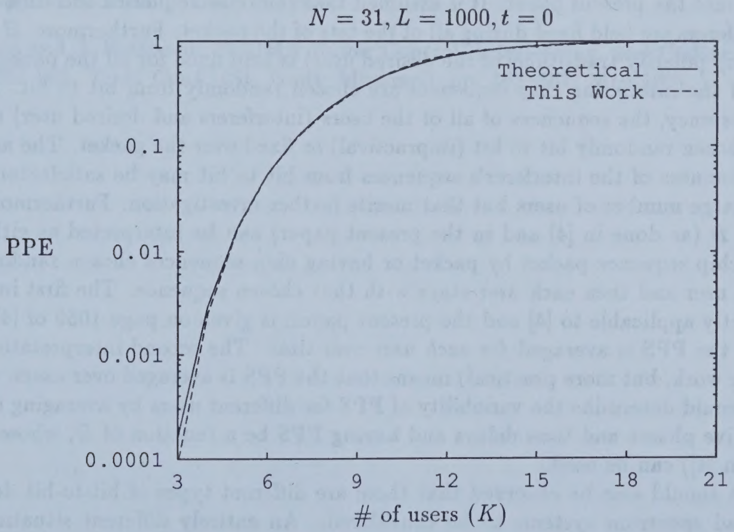


Figure 8: P_E *and* $\hat{P}_E$

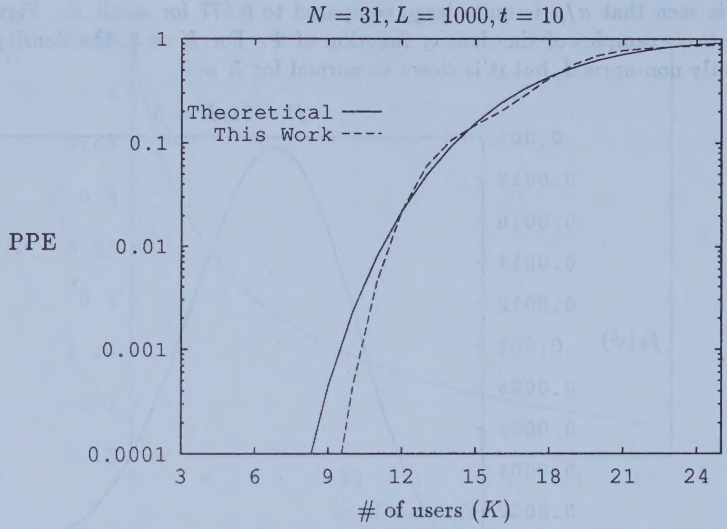


Figure 9: P_E *and* $\hat{P}_E$

VI. Concluding Remarks

This paper presented an approximation for the probability of packet success based on the work of [4]. We obtained an improvement in simplicity and accuracy with the final calculations.

To interpret the results, we repeat some key assumptions made in [4], and carried over into the present paper. It is assumed that the relative phases and time delays of the interferers are held fixed during all of the bits of the packet. Furthermore, B (the number of chip polarity transitions of the desired user) is held fixed for all the packet's bits while all of the interfering chip sequences are chosen randomly from bit to bit. Actually, for consistency, the sequences of all of the users (interferers and desired user) should either be chosen randomly bit to bit (impractical) or fixed over the packet. The assumption of randomness of the interferer's sequences from bit to bit may be satisfactory because of the large number of users but that merits further investigation. Furthermore, averaging over B (as done in [4] and in the present paper) can be interpreted as either changing the chip sequence packet by packet or having chip sequences chosen randomly once for each user and then each user stays with that chosen sequence. The first interpretation, directly applicable to [4] and the present paper, is given on page 1059 of [4], and means that the PPS is averaged *for each user* over time. The second interpretation (requiring more work, but more practical) means that the PPS is averaged over users. In that case, one could determine the variability of PPS for different users by averaging only over the relative phases and time delays and having PPS be a function of B, whose distribution (from [4]) can be used.

It should also be observed that there are different types of bit-to-bit dependence in spread spectrum systems to be considered. An entirely different situation is that of residual correlations due in part to finite interleaving in a fading environment (see, e.g., [8] and references therein).

References

[1] M. B. Pursley, "Performance Evaluation for Phase-Coded Spread-Spectrum Multiple Access Communication–Part I: System Analysis," *IEEE Trans. on Commun.*, vol. COM-25, pp. 795–799, Aug. 1977.

[2] E. A. Geraniotis and M. B. Pursley, "Error Probability for Direct-Sequence Spread-Spectrum Multiple-Access Communications–Part II: Approximations," *IEEE Trans. on Commun.*, vol. COM-30, pp. 985–995, May 1982.

[3] J. S. Lehnert and M. B. Pursley, "Error Probabilities for Binary Direct-Sequence Spread-Spectrum Communications with Random Signature Sequences," *IEEE Trans. on Commun.*, vol. COM-35, pp. 87–98, Jan. 1987.

[4] R. K. Morrow Jr. and J. S. Lehnert, "Bit-to-Bit Error Dependence in Slotted DS/SSMA Packet Systems with Random Signature Sequences," *IEEE Trans. on Commun.*, vol. COM-37, no. 10, pp. 1052–1061, Oct. 1989.

[5] M. B. Pursley, D. V. Sarwate, and W. E. Stark, "Error Probability for Direct-Sequence Spread-Spectrum Multiple-Access Communications–Part I: Upper and Lower Bounds," *IEEE Trans. on Commun.*, vol. COM-30, pp. 975–984, May 1982.

[6] K. Yao, "Error Probability of Asynchronous Spread Spectrum Multiple Access Communication Systems," *IEEE Trans. on Commun.*, vol. COM-25, no. 8, pp. 803–809, Aug. 1977.

[7] J. M. Holtzman, "A Simple, Accurate Method to Calculate Spread Spectrum Multiple Access Error Probabilities," in *Proc. Int. Conf. Commun.*, (Denver, Colorado), June 1991, to appear in IEEE Trans. Commun.

[8] F. Simpson and J. Holtzman, "CDMA Power Control, Interleaving, and Coding," in *Proc. IEEE Veh. Tech. Conf.*, (St. Louis, Missouri), pp. 362–367, May 1991.

ASYMPTOTIC ANALYSIS OF DIRECT-SEQUENCE SPREAD-SPECTRUM MULTIPLE-ACCESS COMMUNICATION SYSTEMS

J. S. Lehnert
School of Electrical Engineering
Purdue University
West Lafayette, IN 47907-1285

Abstract

Asymptotic results have been obtained for systems with binary direct-sequence spread-spectrum (DS/SS) signaling, an asynchronous multiple-access channel, and coherent correlation receivers. The results allow accurate evaluation of the performance of DS/SS multiple-access systems for both large and small numbers of simultaneous users with significantly less computation than competing methods and much more accuracy than the standard Gaussian approximation.

Introduction

In this paper we are concerned with binary direct-sequence spread-spectrum (DS/SS) signaling, an asynchronous multiple-access channel, and a coherent correlation receiver. We assume that all the signature sequences are randomly generated. Each of K transmitters is assumed to have the same power, although the general approach can accommodate the case of transmitters with unequal powers. The system has been precisely defined, and an expression for the characteristic function of the decision statistic of the correlation receiver has been found [1]. Here, we examine the asymptotic behavior of this decision statistic and the corresponding characteristic function as the number N of spread-spectrum chips per transmitted data bit becomes large. Although the number of sequence combinations to consider in order to evaluate the error probability is 2^{KN}, the computational complexity can be reduced significantly [1]. The asymptotic results we develop here are shown to adequately model systems with N as small as 31. Even for large K and large N, the average error probability can be closely approximated with the following results in a few minutes.

System description

The model under consideration is detailed in [1]. Each transmitter-receiver pair in the model encodes and decodes data using a particular sequence. We assume, however, that instead of carefully choosing a signature sequence for each transmitter-receiver pair, a signature sequence has been chosen at random from the set of all possible sequences. Each sequence in the set of all possible sequences has an equal probability of being chosen and the choice of a sequence for each transmitter-receiver pair is made independently of the choices made for all other pairs. The average probability of error that we obtain is an average with respect to all the possible combinations of signature sequences that might be selected. The decision statistic for the coherent correlation receiver for the first transmitter is comprised of a contribution from the first transmitter and a contribution from K-1 interfering transmitters. Here, we assume that the contribution of thermal noise is negligible when compared to the contribution from the multiple-access interference.

After appropriate normalization so that the signal contribution from the first transmitter has unit magnitude, the characteristic function of the interference Ξ at the first correlation receiver for a system employing a rectangular chip waveform can be expressed as [1]

$$\Phi_{\Xi}(u) = 2^{1-N}\sum_{i=0}^{N-1} C(N-1, i)\left\{ \frac{4}{\pi T_c} \int_0^{\frac{\pi}{2}} \int_0^{\frac{T_c}{2}} z(u\,;\, s,\, \phi,\, i,\, N-1-i\,)ds\,d\phi \right\}^{K-1} \tag{1}$$

where

$$z(u\,;\, s,\phi,i,j) = \cos[uT^{-1}(1-s)\cos\phi]\cos[uT^{-1}s\cos\phi]\{\cos[uT^{-1}\cos\phi]\}^{i}\{\cos[uT^{-1}(1-2s)\cos\phi]\}^{j}. \tag{2}$$

In the above equations, the function C(n,k) represents the binomial coefficient $\binom{n}{k}$, and the parameter T represents the duration of a data bit, i.e., $T=NT_c$, where T_c represents the duration of a rectangular chip and N is the number of chips used to spread the spectrum of a data bit.

Systems with large N:

For clarity we specialize (1) to the case K=3 and let the pair of random variables B_2 and B_3 model the interferences from the second and third transmitters, respectively. The random variables s_i and ϕ_i represent the differential chip delay and the differential carrier phase, respectively, of the i–th user for i=2,3. We show that in the limit as N approaches infinity the conditional characteristic function of the interference given s_2, s_3, ϕ_2, and ϕ_3, becomes that of the sum of a pair of independent, jointly Gaussian random variables. The generalization to larger K follows in a straightforward fashion. Setting $T_c=1$ and normalizing the decision statistic by $k_0\sqrt{N}$ for some constant k_0, the characteristic function of the interference $\hat{\Xi}$ can be expressed as

$$\Phi_{\hat{\Xi}}(u) = 2^{1-N}\left(\frac{4}{\pi}\right)^2 \int_0^{\frac{\pi}{2}}\int_0^{\frac{\pi}{2}}\int_0^{\frac{1}{2}}\int_0^{\frac{1}{2}} \gamma(u,s_2,\phi_2)\gamma(u,s_3,\phi_3)\cdot \tag{3}$$

$$\sum_{i=0}^{N-1} C(N-1,\, i)[\alpha(u,\phi_2)\alpha(u,\phi_3)]^{i}[\beta(u,s_2,\phi_2)\beta(u,s_3,\phi_3)]^{N-1-i}ds_2ds_3d\phi_2d\phi_3$$

where

$$\gamma(u,s,\phi)=\cos[uN^{-1/2}k_0(1-s)\cos\phi]\cos[uN^{-1/2}k_0 s\cos\phi]\ ,$$

$$\alpha(u,\phi)=\cos[uN^{-1/2}k_0\cos\phi]\ ,$$

and

$$\beta(u,s,\phi)=\cos[uN^{-1/2}k_0(1-2s)\cos\phi]\ .$$

Applying the Binomial Theorem to (3), we obtain

$$\Phi_{\Xi}(u)=\left(\frac{4}{\pi}\right)^2\int_0^{\frac{\pi}{2}}\int_0^{\frac{\pi}{2}}\int_0^{\frac{1}{2}}\int_0^{\frac{1}{2}}\gamma(u,s_2,\phi_2)\gamma(u,s_3,\phi_3)\cdot \quad (4)$$

$$\left(\frac{\alpha(u,\phi_2)\alpha(u,\phi_3)+\beta(u,s_2,\phi_2)\beta(u,s_3,\phi_3)}{2}\right)^{N-1}ds_2ds_3d\phi_2d\phi_3\ .$$

We now consider the limit of the integrand of (4) for fixed s_2, ϕ_2, s_3, and ϕ_3 as N approaches infinity. The natural logarithm of the integrand is the indeterminate form given by

$$f(u,s_2,\phi_2,s_3,\phi_3,N)=\log[\gamma(u,s_2,\phi_2)]+\log[\gamma(u,s_3,\phi_3)] \quad (5)$$

$$+\frac{\log\left[\dfrac{\alpha(u,\phi_2)\alpha(u,\phi_3)+\beta(u,s_2,\phi_2)\beta(u,s_3,\phi_3)}{2}\right]}{(N-1)^{-1}}\ .$$

Applying L'Hospital's rule to (5) yields the limit of the logarithm of the integrand of (4). The result is

$$\lim_{N\to\infty}f(u,s_2,\phi_2,s_3,\phi_3,N)=-\frac{k_0^2u^2}{2}[(1-2s_2+2s_2^2)\cos^2\phi_2+(1-2s_3+2s_3^2)\cos^2\phi_3]\ . \quad (6)$$

For fixed s_2, ϕ_2, s_3, and ϕ_3, the exponential function with the argument of (6) is the characteristic function of the sum of a pair of independent jointly Gaussian random variables.

Numerical results

The density function for the random variable A_2, where $B_2=A_2\cos\phi_2$ in [1], modeling the interference from the second transmitter given a differential carrier phase ϕ_2 of zero is shown in Fig. 1. For Fig. 1, the decision statistic is normalized so that the contribution from the first transmitter at the first correlation receiver has a magnitude of N. The actual density of the random variable A_2 consists of a component that is piecewise constant and a component consisting of impulses. For a given fixed differential chip delay s_2, the conditional density function of the random variable A_2 modeling the interference from the second transmitter (for $\phi_2=0$) consists of a set of impulses. Hence, it is difficult to compare this conditional density function to a Gaussian density function. To construct Fig. 2, this problem is circumvented by representing the impulses as narrow rectangles with the same area as the weights of the impulses in the correct density function. The resulting density function is then normalized so that the interference (given s_2) has unit variance. For large N, the fixed width of a rectangle used to model an impulse in the conditional density of A_2 approaches zero after normalization, and the approximation becomes reasonable. In Fig. 2, the normalized conditional density function of the interference from the second transmitter (for $\phi_2=0$ and given a fixed differential chip delay s_2) is shown by a solid line. A comparison is made to the density (dotted line) of a zero-mean Gaussian random variable with the same variance.

In the limit as N approaches infinity, the sum of a conditionally independent set of jointly Gaussian random variables adequately models the multiple-access interference for fixed differential delays and phases. The zero-mean sum is itself conditionally Gaussian, and can be characterized by its variance. The distribution of the variance for K active users can be computed when the differential delays and phases are random using the techniques of [2]. The resulting density function [2] $f_\Psi(x)$ of the variance Ψ is the (K−2)–fold convolution of the function $f_Z(x)$ given by

$$f_Z(x)=\frac{1}{\pi\sqrt{2Nx}}\log\left|\frac{\sqrt{(N-x)}+\sqrt{(N/2)}}{\sqrt{(N-x)}-\sqrt{(N/2)}}\right| \tag{7}$$

for $x \neq N/2$ and $0<x\leq N$, given that the decision statistic is normalized so that the magnitude of the contribution from the first transmitter at the first receiver is N.

In Fig. 3, the solid lines depict upper and lower bounds on the actual error probability obtained using the methods of [1] for the case N=1 and a rectangular chip waveform. The standard Gaussian approximation to the average probability of bit error given by Q(SNR), where $Q(x)=\frac{1}{\sqrt{2\pi}}\int_x^\infty e^{-u^2/2}du$ and the argument SNR is the signal-to-noise ratio of the system [3], is shown by a dotted line that lies outside the bounds for K<15. The other dotted line depicts the results obtained by making the conditional Gaussian assumption given the differential chip delays and carrier phases of the interferers and using the equation for the approximate error probability $\hat{P}_e$ given by

$$\hat{P}_e=\int_0^\infty Q\left[\frac{N}{\sqrt{\psi}}\right]f_\Psi(\psi)d\psi\,. \tag{8}$$

Fig. 3 shows that the latter approximation is much more accurate than the former for small numbers of interferers. The value $\hat{P}_e$ in (8) converges to the correct average error probability as N approaches infinity.

Conclusions

The multiple-access interference in a DS/SS system can be modeled by a set of conditionally independent jointly Gaussian random variables in the limit as N approaches infinity when given the differential chip delays and carrier phases of the interferers. For values of N as small as 31, the expressions that are correct in the limit give an accurate indication of system performance with a very small amount of computation. This is important for evaluating more complex DS/SS systems [4].

Acknowledgement

This work is supported by National Science Foundation Grant CDR8803017 to the Engineering Research Center for Intelligent Manufacturing Systems.

References

[1] J. S. Lehnert and M. B. Pursley, "Error probabilities for binary direct-sequence spread-spectrum communications with random signature sequences," *IEEE Transactions on Communications*, vol. COM-35, pp. 87-98, January 1987.

[2] R. K. Morrow, Jr. and J. S. Lehnert, "Bit to bit error dependence in slotted DS/SSMA packet systems with random signature sequences," *IEEE Transactions on Communications*, to appear October 1989.

[3] M. B. Pursley, "Performance evaluation for phase-coded spread-spectrum multiple-access communication --Part I: System analysis," *IEEE Transactions on Communications* , vol. COM-25, no. 7, pp. 795-799, August 1977.

[4] G. L. Turin, "Introduction to spread-spectrum antimultipath techniques and their application to urban digital radio," *Proceedings of the IEEE* , vol. 68, no. 3, March 1980.

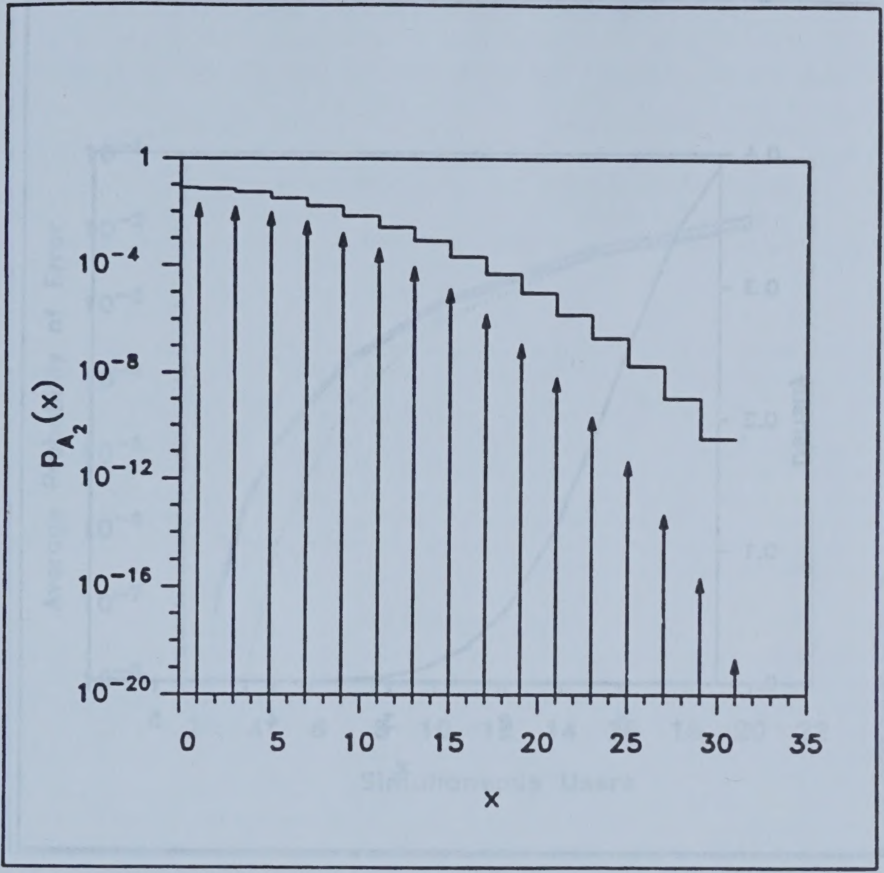


Fig. 1. Density function for the random variable A_2 (N=31, rectangular chip waveform).

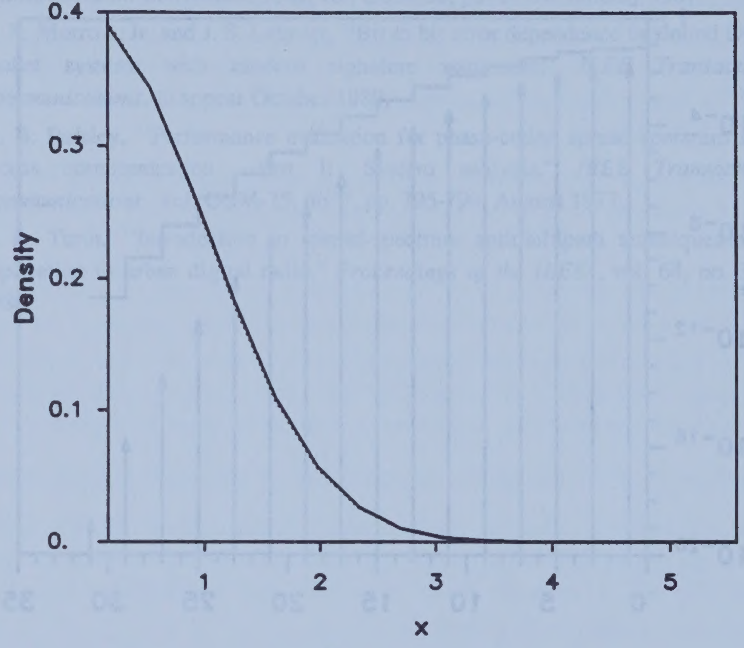


Fig. 2. Density function after normalization for the random variable A_2 given s_2 compared to a Gaussian density function (N=31, rectangular chip waveform, impulses represented by rectangles).

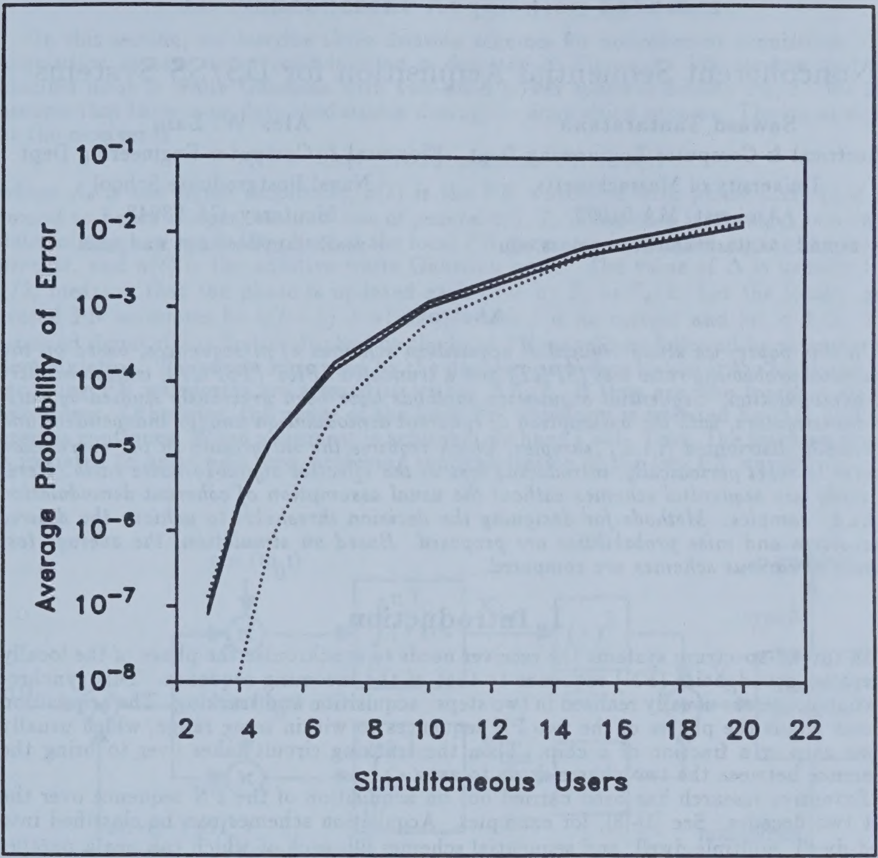


Fig. 3. Bounds on the average probability of error vs. the number of simultaneous users compared to approximations.

Noncoherent Sequential Acquisition for DS/SS Systems

Sawasd Tantaratana
Electrical & Computer Engineering Dept
University of Massachusetts
Amherst, MA 01003
e-mail: tantaratana@ecs.umass.edu

Alex W. Lam
Electrical & Computer Engineering Dept
Naval Postgraduate School
Monterey, CA 93943
e-mail: lam@ece.nps.navy.mil

Abstract

In this paper, we study sequential acquisition schemes of m-sequences, based on the sequential probability ratio test (SPRT) and a truncated SPRT (TSPRT), with noncoherent demodulation. Sequential acquisition methods have been previously studied by various investigators, with the assumption of coherent demodulation and/or independent and identically distributed (i.i.d.) samples, which requires the integrators in the correlation process to reset periodically, introducing loss in the effective signal-to-noise ratio. Here, we study two sequential schemes without the usual assumption of coherent demodulation or i.i.d. samples. Methods for designing the decision thresholds to achieve the desired false-alarm and miss probabilities are proposed. Based on simulation, the average test lengths of various schemes are compared.

I. Introduction

In spread-spectrum systems the receiver needs to synchronize the phase of the locally generated pseudonoise (PN) sequence to that of the incoming sequence. This synchronization process is usually realized in two steps: acquisition and tracking. The acquisition process aligns the phases of the two PN sequences to within some range, which usually is one chip or a fraction of a chip. Then the tracking circuit takes over to bring the difference between the two phases down to zero.

Extensive research has been carried out on acquisition of the PN sequence over the past two decades. See [1]-[8], for examples. Acquisition schemes can be classified into fixed-dwell, multiple-dwell, and sequential schemes [9], each of which can use a parallel or serial search strategy. Sequential schemes are least studied because of the difficulty in design and analysis. All reports on sequential schemes assume either that coherent demodulation is available or that the samples obtained for the decision process are independent and identically distributed (i.i.d.). To obtain i.i.d. samples, the integrators in the acquisition system have to be periodically reset, which in effect reduces the signal-to-noise ratio.

In this paper, we study *noncoherent* sequential acquisition schemes using serial search for direct-sequence spread-spectrum (DS/SS) systems. We use the sequential probability ratio test (SPRT) and a truncated SPRT for deciding whether the local sequence is in alignment with the incoming sequence. The samples obtained from the noncoherent demodulator are not i.i.d. Methods are presented for designing the thresholds of the SPRT and the TSPRT such that the desired false-alarm and detection probability are satisfied. The average test lengths are obtained through simulations and compared to the noncoherent fixed-dwell acquisition scheme.

This work was supported by the Army Research Office under Grant DAAL03-91-G-0001.

II. Noncoherent Acquisition Schemes

In this section, we describe three decision schemes for noncoherent acquisition. The acquisition system under consideration is depicted in Figure 1. We assume that the channel noise is white Gaussian with two-sided power spectral density $N_0/2$. We also assume that there is no data modulation during the acquisition process. The input signal at the receiver is

$$r(t) = A_0 a(t + i\Delta T_c)\cos(\omega_0 t + \theta) \; + \; n(t) \tag{1}$$

where A_0 is the signal amplitude, $a(t)$ is the PN waveform with phase $i\Delta T_c$ (i is assumed to be an integer, without loss of generality), T_c is the chip interval, Δ is a value determining how much the phase of the local PN generator is updated during the search process, and $n(t)$ is the additive white Gaussian noise. The value of Δ is usually 1 or 1/2, meaning that the phase is updated each time by T_c or $T_c/2$. Let the locally generated PN waveform be $a(t + (j + \gamma)\Delta T_c)$, where j is an integer and $|\gamma| < 1/2$. The received signal $r(t)$ is first multiplied by the local PN waveform, followed by noncoherent demodulation. The result Y_n is used by the decision processor to test if the local and the incoming PN waveforms are aligned to within $\Delta T_c/2$. If they are, the tracking circuit takes over. Otherwise, the phase of the local PN waveform is updated by ΔT_c and the process continues. When alignment is achieved, we have $j = i$. Then, the tracking circuit is initiated to bring γ to zero, achieving synchronization of the two PN waveforms.

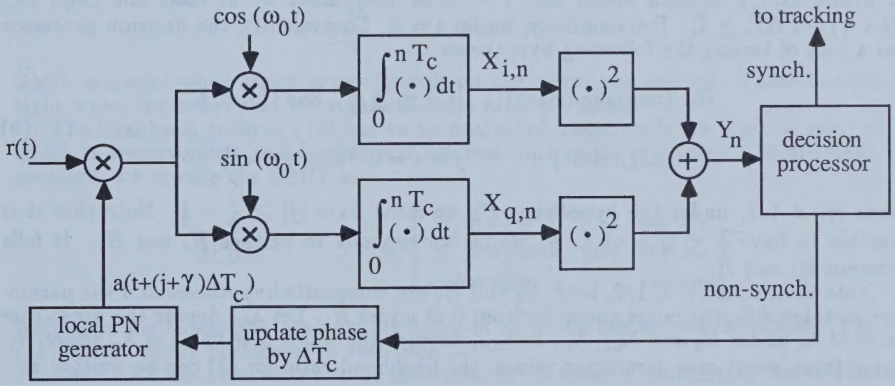


Figure 1 Block diagram of noncoherent serial acquisition scheme

Referring to Figure 1, the inphase and the quadrature components, after integration over n chips, i.e., from $t = 0$ to nT_c, and ignoring the double-frequency terms, are

$$X_{i,n} = \int_0^{nT_c} r(t)a(t + (j + \gamma)\Delta T_c)\cos(\omega_0 t)\, dt = \frac{A_0}{2} T_c S_n \cos\theta \; + \; N_{i,n} \tag{2}$$

$$X_{q,n} = \int_0^{nT_c} r(t)a(t + (j + \gamma)\Delta T_c)\sin(\omega_0 t)\, dt = \frac{A_0}{2} T_c S_n \sin\theta \; + \; N_{q,n} \tag{3}$$

where

$$N_{i,n} = \int_0^{nT_c} n(t)a(t + (j + \gamma)\Delta T_c)\cos(\omega_0 t)dt \tag{4}$$

$$N_{q,n} = \int_0^{nT_c} n(t)a(t + (j + \gamma)\Delta T_c)\sin(\omega_0 t)dt \tag{5}$$

are independent Gaussian random variables with mean zero and variance $\sigma_n^2 = nT_cN_0/4$, and where

$$S_n = \frac{1}{T_c}\int_0^{nT_c} a(t+i\Delta T_c)a(t+(j+\gamma)\Delta T_c)dt \tag{6}$$

Note that $N_{i,n}$, $N_{q,n}$, and S_n depend on the length of integration. For a fixed n, $X_{i,n}$ and $X_{q,n}$ are independent Gaussian random variables with variance σ_n^2 and means $(A_0/2)T_cS_n\cos\theta$ and $(A_0/2)T_cS_n\sin\theta$, respectively. Note, however, that $X_{i,n}$ and $X_{i,m}$ are not independent for $n \neq m$, and similarly for $X_{q,n}$ and $X_{q,m}$. The test statistic for deciding alignment or non-alignment is

$$Y_n = X_{i,n}^2 + X_{q,n}^2 \tag{7}$$

The random variable Y_n has a Chi-square probability density function (pdf) with two degrees of freedom, given by

$$f_{Y_n}(y_n) = \frac{1}{2\sigma_n^2}e^{-(y_n+\lambda_n)/2\sigma_n^2}\, I_0(\frac{\sqrt{\lambda_n y_n}}{\sigma_n^2}), \quad y_n \geq 0 \tag{8}$$

where $\lambda_n = [(A_0/2)T_cS_n\cos\theta]^2 + [(A_0/2)T_cS_n\sin\theta]^2 = (A_0^2/4)T_c^2S_n^2$, and where $I_0(.)$ is the modified Bessel function of order zero.

Based on Y_n, the decision processor must decide whether the two phases are aligned to within $\Delta T_c/2$ of each other, i.e., $j = i$, or they differ by at least one chip, i.e., $|(j+\gamma) - i|\Delta T_c \geq T_c$. For simplicity, we let $i = 0$. Equivalently, the decision processor has a task of testing the following hypotheses

$$\begin{aligned} &H_0 \text{ (non-alignment)} : && |j+\gamma| \geq \tfrac{1}{\Delta} \quad \text{and} \quad |\gamma| < \tfrac{1}{2} \\ &H_1 \text{ (alignment)} : && j = i = 0 \quad \text{and} \quad |\gamma| < \tfrac{1}{2} \end{aligned} \tag{9}$$

Since $|\gamma| < 1/2$, under the hypothesis H_0 we must have $|j| \geq \frac{1}{\Delta} - \frac{1}{2}$. Note that it is possible to have $\frac{1}{2} < |j+\gamma| < \frac{1}{\Delta}$, which corresponds to neither H_0 nor H_1. It falls between H_0 and H_1.

Note that, since $|\gamma| < 1/2$, both H_0 and H_1 are composite hypotheses and the parameter λ_n takes different range under H_0 from that under H_1. Let $\lambda_{n,0}$ denote the worst-case value of λ_n under H_0 and $\lambda_{n,1}$, $\lambda_{n,1} > \lambda_{n,0}$, denotes the worst-case value of λ_n under H_1. Using these worst-case parameter values, the likelihood ratio for (9) can be written as

$$\Lambda_n(y_n) = \frac{f_{Y_n}(y_n|H_1)}{f_{Y_n}(y_n|H_0)} = \exp(\frac{\lambda_{n,0}-\lambda_{n,1}}{2\sigma_n^2})\frac{I_0(\sqrt{(\lambda_{n,1}/\sigma_n^2)(y_n/\sigma_n^2)})}{I_0(\sqrt{(\lambda_{n,0}/\sigma_n^2)(y_n/\sigma_n^2)})} \tag{10}$$

where y_n is a realization of Y_n. Next, we define the fixed-dwell, sequential, and truncated sequential decision schemes, based on the likelihood ratio (10).

In a fixed-dwell (single-dwell) or fixed-sample-size (FSS) decision scheme, the length of the integration is fixed *a priori* and a decision is based on the resulting test statistic. If the integration length is from $t = 0$ to MT_c, the FSS test compares $\Lambda_M(y_M)$ to a threshold τ. It can be shown that the likelihood ratio (10) is a monotonically increasing function of the variable, provided that $\lambda_{n,1} > \lambda_{n,0}$, which is the case in our situation. Therefore, an equivalent test is

$$\underline{\text{FSS:}} \qquad y_M \begin{cases} \geq \tau' = \Lambda_M^{-1}(\tau) & \text{say } H_1 \\ < \tau' & \text{say } H_0 \end{cases} \tag{11}$$

where $\Lambda_n^{-1}(.)$ is the inverse function of $\Lambda_n(.)$.

A sequential probability ratio test (SPRT) is obtained by testing the likelihood ratio against two thresholds for $n = 1, 2, ...$ until the upper threshold or the lower threshold is reached. Note that the length of integration increases by one chip each time n increases by one. H_1 is accepted if the upper threshold is reached first, and H_0 is accepted if the lower threshold is reached first. The test can be described as

$$\underline{\text{SPRT:}} \qquad \Lambda_n(y_n) \begin{cases} \geq A & \text{say } H_1 \\ \leq B & \text{say } H_0 \\ \text{otherwise,} & \text{continue to next } n \end{cases} \tag{12}$$

Note that Y_n and Y_{n+1} are not independent and that the likelihood ratio $\Lambda_n(Y_n)$ cannot be written as the sum of independent random variables. However, the Type I error (false-alarm probability) and the Type II error (miss probability) can still be related through Wald's inequalities [10], namely,

$$A \leq \frac{1-\beta}{\alpha} \qquad \text{and} \qquad B \geq \frac{\beta}{1-\alpha} \tag{13}$$

where α and β are the resulting false-alarm and miss probabilities, respectively. If the excess over the boundary is small when the test terminates, which is the case when the average test length is large, the inequalities can be approximated by the equalities $A \approx (1-\beta)/\alpha$ and $B \approx \beta/(1-\alpha)$. Since $1-\beta < 1$ and $1-\alpha < 1$, (13) implies

$$\alpha < \frac{1}{A} \qquad \text{and} \qquad \beta < B \tag{14}$$

which is useful when strict upper bounds on the errors are desired. These bounds are tight when the errors and the signal-to-noise ratio are small.

The likelihood ratio in (12) has to be evaluated numerically for a given value of y_n, which is inconvenient and impractical for real-time implementation. To alleviate this problem, we rewrite the SPRT as

$$\underline{\text{SPRT:}} \qquad y_n \begin{cases} \geq A(n) \equiv \Lambda_n^{-1}(A) & \text{say } H_1 \\ \leq B(n) \equiv \Lambda_n^{-1}(B) & \text{say } H_0 \\ \text{otherwise,} & \text{continue} \end{cases} \tag{15}$$

Note that the thresholds are now functions of n, which can be pre-calculated. The test (15) can be easily performed in real-time.

The SPRT has been shown to be optimum, in the sense of minimizing the average test length under H_0 and H_1 among all tests for a given pair of Type I and Type II errors, only for the case that the samples are i.i.d.[11]. In the case of non-i.i.d., as in our situation here, no optimality has been established. However, it is expected that the performance is still good and much better than the FSS likelihood ratio test. This statement will be confirmed by numerical results.

One drawback of the SPRT is that there is no upper bound on the test length. Therefore, it is possible that the decision processor can be stuck at a particular uncertainty phase for a long period of time, which is undesirable. Such event can occur especially when the difference between the phases of the two PN waveforms is such that it corresponds neither to H_0 nor to H_1, namely, $1/2 < |j+\gamma| < 1/\Delta$. To avoid a lengthy test, we propose to use a truncated SPRT (TSPRT), described by

$$\underline{\text{TSPRT:}} \qquad \text{if } n < \hat{n}, \quad \Lambda_n(y_n) \begin{cases} \geq \hat{A} & \text{say } H_1 \\ \leq \hat{B} & \text{say } H_0 \\ \text{otherwise,} & \text{continue} \end{cases}$$

$$\text{if } n = \hat{n}, \quad \Lambda_n(y_n) = \Lambda_{\hat{n}}(y_{\hat{n}}) \begin{cases} \geq \hat{\tau} & \text{say } H_1 \\ < \hat{\tau} & \text{say } H_0 \end{cases} \tag{16}$$

Therefore, the test is abruptly truncated at $n = \hat{n}$ if it has not previously terminated. The design of the thresholds $\hat{A}$, $\hat{B}$, $\hat{\tau}$, and truncation point $\hat{n}$ will be addressed in the next section. Using the monotonicity of $\Lambda_n(.)$, the TSPRT can be written as

$$\underline{\text{TSPRT:}} \qquad \text{if } n < \hat{n}, \qquad y_n \begin{cases} \geq \hat{A}(n) \equiv \Lambda_n^{-1}(\hat{A}) & \text{say } H_1 \\ \leq \hat{B}(n) \equiv \Lambda_n^{-1}(\hat{B}) & \text{say } H_0 \\ \text{otherwise} & \text{continue} \end{cases}$$

$$\text{if } n = \hat{n}, \qquad y_{\hat{n}} \begin{cases} \geq \Lambda_{\hat{n}}^{-1}(\hat{\tau}) & \text{say } H_1 \\ < \Lambda_{\hat{n}}^{-1}(\hat{\tau}) & \text{say } H_0 \end{cases} \tag{17}$$

If we compare the time to acquisition of different tests, the test with the smallest expected sample size reaches acquisition the soonest, if all the tests have the same false-alarm and miss probabilities. If the initial phase offset between the incoming and local sequences is uniformly distributed over one period, then the hypothesis H_0 is most likely to occur. Therefore, the expected sample size under H_0 contributes the most to the computation of the expected acquisition time. Hence, it is best to minimize the expected sample size under H_0 while keeping the expected sample sizes under H_1 and under the case between H_0 and H_1 to within tolerable values.

Note that in the sequential schemes reported in [5], [6], and [9], the integrators for the inphase and quadrature components are reset after every chip or every few chips, in order to obtain independent (or nearly independent) and identically distributed samples. Their purpose is to make the formulas obtained by Wald [10] applicable. However, resetting the integrator induces loss in the effective signal-to-noise ratio at the decision processor (noncoherent combining loss). The SPRT (12) and TSPRT (16) proposed here do not reset the integrator, generating dependent and non-identically distributed samples. This means that we need design methods such that the decision processor can perform as desired. In the next section, methods are described for designing the thresholds of the SPRT (12) and the TSPRT (16). Performances of these tests are presented in Section IV using simulation.

III. Design of Decision Processors

Assume that an m-sequence is used for the PN sequence. Denote the chips of the PN sequence by c_k, $k = ..., 0, 1, 2, ...$, where $c_k = \pm 1$, and let the period be $N = 2^m - 1$, so that $c_{k+N} = c_k$ for all k. The PN waveform is

$$a(t) = \sum_{k=-\infty}^{\infty} c_k p_{T_c}(t - kT_c) \tag{18}$$

where $p_{T_c}(t) = 1$ for $0 \leq t < T_c$ and it is zero elsewhere. Assuming that $i = 0$, expressions for S_n can be obtained from (6) and (18), yielding

$$\begin{aligned} S_n &= \begin{cases} \sum_{k=0}^{n-1} c_k[(1 - |\gamma|\Delta)c_k + (|\gamma|\Delta)c_{k+\text{sgn}(\gamma)}] & \text{under } H_1 \\ \sum_{k=0}^{n-1} c_k[(1 - \delta)c_{k+\ell} + \delta c_{k+1+\ell}] & \text{under } H_0 \end{cases} \\ &= \begin{cases} n(1 - |\gamma|\Delta) + (|\gamma|\Delta)\sum_{k=0}^{n-1} c_k c_{k+\text{sgn}(\gamma)} \equiv S_{n,1} & \text{under } H_1 \\ (1 - \delta)\sum_{k=0}^{n-1} c_k c_{k+\ell} + \delta \sum_{k=0}^{n-1} c_k c_{k+1+\ell} \equiv S_{n,0} & \text{under } H_0 \end{cases} \end{aligned} \tag{19}$$

where $\text{sgn}(x)$ is 1 for $x \geq 0$ and -1 for $x < 0$, $\ell = \lfloor (j+\gamma)\Delta \rfloor$, and $\delta = (j+\gamma)\Delta - \lfloor (j+\gamma)\Delta \rfloor$. Here, $\lfloor x \rfloor$ denotes the integer part of x. Note that $\ell \neq 0$ or -1 and that $0 \leq \delta < 1$.

Defining the (per-chip) signal-to-noise ratio as $\text{SNR} = A_0^2 T_c/(2N_0)$, we have

$$\frac{\lambda_n}{2\sigma_n^2} = \begin{cases} \frac{1}{n}(\text{SNR})S_{n,1}^2 & \text{under } H_1 \\ \frac{1}{n}(\text{SNR})S_{n,0}^2 & \text{under } H_0 \end{cases} \tag{20}$$

Since the receiver does not know the exact value of $S_{n,0}$ and $S_{n,1}$, some nominal values for the worst case must be used in designing the test. Results from simulations for m-sequences suggest that modeling $\sum_{k=0}^{n-1} c_k c_{k+\mathrm{sgn}(\gamma)} \approx 0$ under H_1, and modeling $\sum_{k=0}^{n-1} c_k c_{k+\ell}$ and $\sum_{k=0}^{n-1} c_k c_{k+1+\ell} \lessapprox \sqrt{n}$ under H_0 yield the desired results. With these, we have

$$\frac{\lambda_n}{2\sigma_n^2} \begin{cases} \approx n(\mathrm{SNR})(1-|\gamma|\Delta)^2 & \equiv \frac{\lambda_{n.1}}{2\sigma_n^2} \quad \text{under } H_1 \\ \lessapprox \mathrm{SNR} & \equiv \frac{\lambda_{n.0}}{2\sigma_n^2} \quad \text{under } H_0 \end{cases} \tag{21}$$

Note that the equality and inequality above are approximates. In the design methods to follow, we use these nominal worst-case values.

The cumulative distribution function (cdf) of Y_n can be written in terms of the Q function as

$$F_{Y_n}(y_n) = P(Y_n \le y_n) = 1 - Q(\sqrt{\lambda_n/\sigma_n^2}, \sqrt{y_n/\sigma_n^2}), \quad y_n \ge 0 \tag{22}$$

where $Q(\zeta, \xi) = \int_\xi^\infty x e^{-(x^2+\zeta^2)/2} I_0(\zeta x)\, dx$ An iterative algorithm for calculating the Q function is given in [12].

We now describe design methods for the thresholds of the FSS test, the SPRT, and the TSPRT, such that the decision errors are within the desired limits. We use α_d and β_d to denote the desired false-alarm and miss probabilities.

Consider the FSS test (11). Using the Q function, we can write the false-alarm and miss probabilities as

$$P_{fa} = Q(\sqrt{\lambda_{M,0}/\sigma_M^2}, \sqrt{\tau'/\sigma_M^2}) \quad \text{and} \quad P_{miss} = 1 - Q(\sqrt{\lambda_{M,1}/\sigma_M^2}, \sqrt{\tau'/\sigma_M^2}) \tag{23}$$

From these, the values of τ' and M can be obtained numerically, such that $P_{fa} \le \alpha_d$ and $P_{miss} \le \beta_d$.

For the SPRT (12), we can use the followings to calculate the thresholds.

$$A = \frac{1}{\alpha_d}, \qquad \text{and} \qquad B = \beta_d \tag{24}$$

From (14), we can see that the resulting false-alarm and miss probabilities are $P_{fa} \le 1/A = \alpha_d$ and $P_{miss} \le B = \beta_d$.

Next, consider the TSPRT (16) or, equivalently, (17). Suppose that the SPRT with thresholds $\hat{A}$ and $\hat{B}$ has errors α_{sprt} and β_{sprt}, and a FSS likelihood ratio test with sample size $\hat{n}$ and threshold $\hat{\tau}$ has errors α_{fss} and β_{fss}. It is shown in the Appendix that the errors of the TSPRT (17) are bounded by

$$\alpha_{tsprt} \le \alpha_{sprt} + \alpha_{fss} \quad \text{and} \quad \beta_{tsprt} \le \beta_{sprt} + \beta_{fss} \tag{25}$$

Therefore, to obtain $\alpha_{tsprt} \le \alpha_d$ and $\beta_{tsprt} \le \beta_d$, we can split these errors into the sums of the errors due to the SPRT and the errors due to the FSS test, namely, we let

$$\alpha_d = p_0\alpha_d + (1-p_0)\alpha_d \quad \text{and} \quad \beta_d = p_1\beta_d + (1-p_1)\beta_d \tag{26}$$

where p_0 and p_1 are some constants in [0,1]. Letting $\alpha_{sprt} = p_0\alpha_d$ and $\beta_{sprt} = p_1\beta_d$, we can design the thresholds $\hat{A}$ and $\hat{B}$ according to (24), namely, $\hat{A} = 1/[p_0\alpha_d]$ and $\hat{B} = p_1\beta_d$. Letting $\alpha_{fss} = (1-p_0)\alpha_d$ and $\beta_{fss} = (1-p_1)\beta_d$, we can design $\hat{\tau}$ and $\hat{n}$ using (23). The resulting TSPRT will have $\alpha_{tsprt} \le \alpha_d$ and $\beta_{tsprt} \le \beta_d$, according to (25)-(26). Note that if $p_0 = p_1 = 0$, the TSPRT becomes the FSS test since $\hat{A} = \infty$ and $\hat{B} = 0$. On the other hand, if $p_0 = p_1 = 1$, the TSPRT becomes the SPRT since $\hat{n} = \infty$. For values of p_0 and p_1 in between, the TSPRT can be viewed as a mixture of the SPRT and the FSS test.

Therefore, the choices of p_0 and p_1 control the degree of mixture between the SPRT and the FSS test.

We have addressed the design problem so that the resulting tests have errors within the desired values. The average test lengths of the SPRT and the TSPRT, however, are difficult to obtain. We do not have expressions for these. Therefore, in Section IV we use simulations to compare the average test lengths for various tests with the same desired errors.

IV. Performance Comparison

In this section, we present numerical results on the expected sample size and the power function for the FSS test, the SPRT, and the TSPRT. The expected sample size, also called average sample number (ASN), is the average number of chips that the test in the decision processor terminates and accepts H_0 or H_1. For the FSS test, the sample size M is deterministic and, therefore, its expected value is equal to M. The power function is the probability of accepting H_1. The results are obtained from simulations. For all the results, the updating of the local PN sequence generator is half a chip, i.e., $\Delta = 1/2$. The per-chip SNR is -10dB. The value of γ used for the design is 0.5. The m-sequence used for the simulations has a length of 1023, with primitive polynomial $1 + x^2 + x^5 + x^6 + x^{10}$. Although results for only one m-sequence are presented, the conclusions hold for m-sequences in general.

Figures 2 and 3 depict the ASN and power functions of the FSS test, SPRT, and TSPRT with $\alpha_d = \beta_d = 0.01$ and with $\alpha_d = 0.001$ and $\beta_d = 0.005$. The TSPRT is designed with $p_0 = p_1 = 0.5$. The functions are plotted in terms of $|j + \gamma|$. When $|j + \gamma| = 0.5$, we have the hypothesis H_1 (since $j = 0$ and $\gamma = 0.5$). The hypothesis H_0 corresponds to $|j + \gamma| \geq 2.0$ and it is represented by $|j + \gamma| = 2.0$ in the figures. The sample size M of the FSS test is 258 in Figure 2 and 360 in Figure 3. Result for the FSS test is obtained by calculation, while those for the SPRT and TSPRT are obtained from simulations. For all tests, the resulting errors are equal to or slightly less than the design values of α_d and β_d. This means that the model (21) works well. We can see from the figure that the ASN of the SPRT is the smallest among the tests under H_0 and H_1. Both the SPRT and the TSPRT perform much better than the FSS test under H_0 and H_1. However, the ASN of the SPRT can be larger than that of the TSPRT for some range of $|j + \gamma|$ between 0.5 and 2.0. It can even be larger than the sample size of the FSS test, especially when the errors α and β are small, such as the case in Figure 3. This range of $|j + \gamma|$ occurs for the phase offset next to the case of synchronization. This can be undesirable for the SPRT since it does not have an upper bound on the sample size. Consequently, the test can last for a long period of time before it eventually stops. This was found to occur during the simulations. The TSPRT, on the other hand, prevents such lengthy test since the sample size is upper bounded by $\hat{n}$. For the cases in Figures 2 and 3, the value of $\hat{n}$ is 306 and 409, respectively.

Among all the uncertainty phases to be searched, almost all of them correspond to the non-synchronization condition (H_0). This implies that the ASN under H_0 affects the expected acquisition time the most. To reduce the expected acquisition time, we should minimize the ASN under H_0. This can be achieved by the TSPRT by letting p_1 be large (close to 1). To see this, we plot the ASN and the power functions of the SPRT and the TSPRT designed under different p_0 and p_1 in Figure 4, with $\alpha_d = 0.001$ and $\beta_d = 0.005$. The TSPRT is designed with $p_0 = 0.1$ and $p_1 = 0.9$, and $p_0 = 0.3$ and $p_1 = 0.9$. We can see that the ASN of the TSPRT under H_0 (represented by $|j + \gamma| = 2.0$ on the graph) is essentially the same as that of the SPRT. The variation of p_0 has an effect on the ASN under H_1, a smaller p_0 produces a larger ASN under H_1. However, a smaller p_0 yields a smaller truncation point $\hat{n}$.

V. Conclusion

We have studied nonceherent acquisition of PN sequences for DS/SS systems. Sequential probability ratio test (SPRT) and truncated sequential probability ratio test (TSPRT) are used for testing whether or not the incoming and the local sequences are in synchronization. The proposed tests use the test statistic which is different from previous studies, that in this paper there is no noncoherent combining loss. Consequently, the test statistic is dependent and non-identical from one chip to the next. Design methods for the thresholds and the stopping point is proposed for the SPRT and the TSPRT. The TSPRT can be designed to be essentially as efficient as the SPRT under non-synchronization between the two sequences, and slightly less efficient than the SPRT under synchronization. The TSPRT is found to be uniformly better than the fixed-dwell scheme.

Appendix - Proof of Equation (25)

Denote by V the probability that the untruncated SPRT with boundaries $\hat{B}$ and $\hat{A}$ terminates after $\hat{n}$ samples at the lower threshold $\hat{B}$, with $\hat{\tau} \leq \Lambda_{\hat{n}}(Y_{\hat{n}}) < \hat{A}$, given that H_0 is true. In another word, $V = \sum_{n=\hat{n}+1}^{\infty} P_{H_0}\{\hat{B} < \Lambda_i(Y_i) < \hat{A}$ for all $i < n$, $\hat{\tau} \leq \Lambda_{\hat{n}}(Y_{\hat{n}}) < \hat{A}$, $\Lambda_n(Y_n) \leq \hat{B}\}$, where $P_{H_0}(\cdot)$ denotes the probability when H_0 is true. The resulting false alarm probability is

$$\begin{aligned}
\alpha_{tsprt} &= \sum_{n=1}^{\hat{n}} P_{H_0}\{\hat{B} < \Lambda_i(Y_i) < \hat{A} \text{ for } i < n, \Lambda_n(Y_n) \geq \hat{A}\} \\
&\qquad + P_{H_0}\{\hat{B} < \Lambda_i(Y_i) < \hat{A} \text{ for } i < \hat{n}, \hat{\tau} \leq \Lambda_{\hat{n}}(Y_{\hat{n}}) < \hat{A}\} \\
&= \sum_{n=1}^{\hat{n}} P_{H_0}\{\hat{B} < \Lambda_i(Y_i) < \hat{A} \text{ for } i < n, \Lambda_n(Y_n) \geq \hat{A}\} \\
&\quad + \sum_{n=\hat{n}+1}^{\infty} \Big[P_{H_0}\{\hat{B} < \Lambda_i(Y_i) < \hat{A} \text{ for } i < n, \Lambda_n(Y_n) \geq \hat{A}, \hat{\tau} \leq \Lambda_{\hat{n}}(Y_{\hat{n}}) < \hat{A}\} \\
&\qquad\qquad + P_{H_0}\{\hat{B} < \Lambda_i(Y_i) < \hat{A} \text{ for } i < n, \Lambda_n(Y_n) \leq \hat{B}, \hat{\tau} \leq \Lambda_{\hat{n}}(Y_{\hat{n}}) < \hat{A}\}\Big] \\
&\leq \sum_{n=1}^{\hat{n}} P_{H_0}\{\hat{B} < \Lambda_i(Y_i) < \hat{A} \text{ for } i < n, \Lambda_n(Y_n) \geq \hat{A}\} \\
&\quad + \sum_{n=\hat{n}+1}^{\infty} \Big[P_{H_0}\{\hat{B} < \Lambda_i(Y_i) < \hat{A} \text{ for } i < n, \Lambda_n(Y_n) \geq \hat{A}, \hat{\tau} \leq \Lambda_{\hat{n}}(Y_{\hat{n}}) < \hat{A}\} \\
&\qquad\qquad + P_{H_0}\{\hat{B} < \Lambda_i({}_i) < \hat{A} \text{ for } i < n, \Lambda_n(Y_n) \geq \hat{A}, \hat{B} < \Lambda_{\hat{n}}(Y_{\hat{n}}) < \hat{\tau}\}\Big] \\
&\quad + \sum_{n=\hat{n}+1}^{\infty} P_{H_0}\{\hat{B} < \Lambda_i(Y_i) < \hat{A} \text{ for } i < n, \Lambda_n(Y_n) \leq \hat{B}, \hat{\tau} \leq \Lambda_{\hat{n}}(Y_{\hat{n}}) < \hat{A}\} \\
&= \sum_{n=1}^{\infty} P_{H_0}\{\hat{B} < \Lambda_i(Y_i) < \hat{A} \text{ for } i < n, \Lambda_n(Y_n) \geq \hat{A}\} \\
&\quad + \sum_{n=\hat{n}+1}^{\infty} P_{H_0}\{\hat{B} < \Lambda_i(Y_i) < \hat{A} \text{ for } i < n, \Lambda_n(Y_n) \leq \hat{B}, \hat{\tau} \leq \Lambda_{\hat{n}}(Y_{\hat{n}}) < \hat{A}\} \\
&= \alpha_{sprt} + V \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(A.1)}
\end{aligned}$$

where α_{sprt} is the resulting false alarm probability of the untruncated SPRT with boundaries $\hat{A}$ and $\hat{B}$. For the second equality in the above derivation to hold, we need the assumption that the untruncated SPRT with thresholds $\hat{A}$ and $\hat{B}$ terminates with probability one. Now, V can be upper bounded by

$$\begin{aligned} V &= \sum_{n=\hat{n}+1}^{\infty} P_{H_0}\{\hat{B} < \Lambda_i(Y_i) < \hat{A} \text{ for } i < n, \Lambda_n(Y_n) \leq \hat{B}, \hat{\tau} \leq \Lambda_{\hat{n}}(Y_{\hat{n}}) < \hat{A}\} \\ &\leq P_{H_0}\{\hat{\tau} \leq \Lambda_{\hat{n}}(Y_{\hat{n}}) < \hat{A}\} \quad \leq \quad P_{H_0}\{\hat{\tau} \leq \Lambda_{\hat{n}}(Y_{\hat{n}})\} \quad = \quad \alpha_{fss} \end{aligned} \tag{A.2}$$

where α_{fss} is the resulting false alarm probability of the FSS-LR test with threshold $\hat{\tau}$ and sample size $\hat{n}$. Note that the first inequality above holds because $0 \leq \sum_{n=\hat{n}+1}^{\infty} P_{H_0}\{\hat{B} < \Lambda_i(Y_i) < \hat{A} \text{ for } i < n,\ \Lambda_n(Y_n) \leq \hat{B}\} \leq 1$. Combining (A.1) and (A.2), we have the first inequality in (25).

Similarly, we can show that the probability of miss for the TSPRT is upper bounded by $\beta_{sprt}+\beta_{fss}$.

References

[1] R. Pickholtz, D. Schilling, and L. Milstein, "Theory of spread-spectrum communications - A tutorial," *IEEE Trans. Commun.*, Vol. COM-30, pp.855-884, May 1982.

[2] D.M. DiCarlo and C.L. Weber, "Multiple dwell serial search: performance and application to direct sequence code acquisition," *IEEE Trans. Commun.,* Vol. COM-31, pp. 650-659, May 1983.

[3] A. Polydoros and C.L. Weber, "A unified approach to serial search spread-spectrum code acquisition - Part I: general theory and Part II: a matched-filter receiver," *IEEE Trans. Commun.,* Vol. COM-32, pp. 542-560, May 1984.

[4] D. Hall and C. Weber, "Noncoherent sequential acquisition of DS waveforms," *Proc. MILCOM Conf.*, pp.13.3.1-5, 1986.

[5] D. Hall and C.L. Weber, "Two Methods for Noncoherent Acquisition of DS Waveforms," *Proc. MILCOM Conference*, pp. 13.7.1-13.7.5, 1987.

[6] H. Meyr and G. Polzer, "A simple method for evaluating the probability density function of the sample number for the optimum sequential detector," *IEEE Trans. Commun.,* Vol. COM-35, pp. 99-103, Jan. 1987.

[7] E. Sourour, S.C. Gupta and W. Refai, "Direct Sequence Spread Spectrum Serial Acquisition in a Nonselective and Frequency Selective Rician Fading Channel," *Proc. MILCOM Conference*, pp. 5.7.1-5.7.5, 1990.

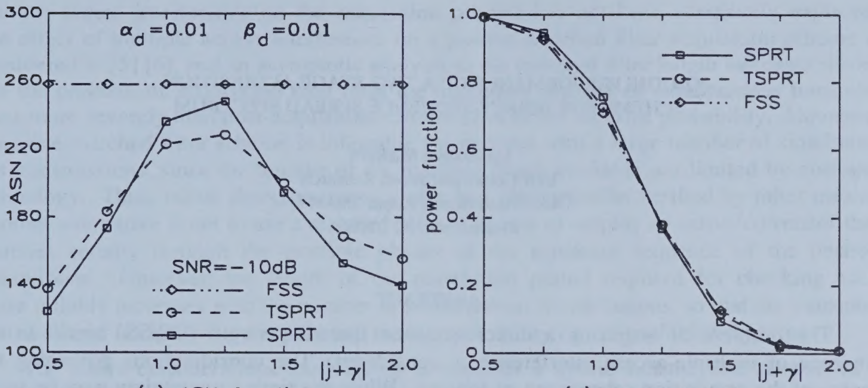
[8] Y.H. Lee and S. Tantaratana, "Acquisition of PN Sequences for DS/SS Systems Using a Truncated Sequential Probability Ratio Test," *Journal of the Franklin Institute,* Vol. 328, pp. 231-248, 1991.

[9] K. Simon, J. Omura, R. Scholtz, and K. Levitt, *Spread Spectrum Communication*, Volume III, Computer Science Press, 1985.

[10] A. Wald, *Sequential Analysis*, John Wiley and Sons, New York, 1947.

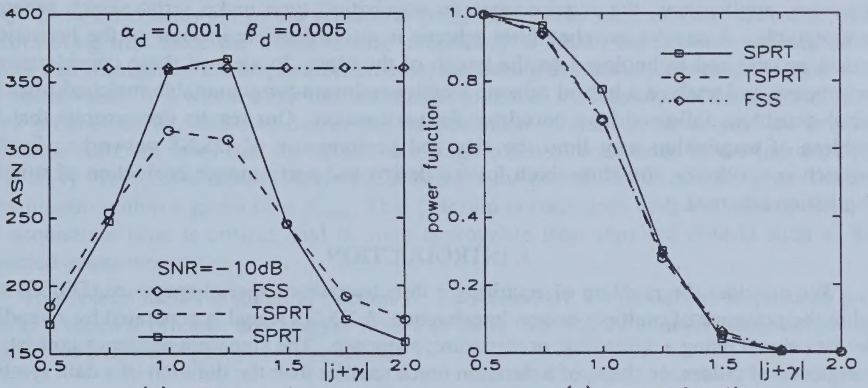
[11] E.L. Lehmann, *Testing Statistical Hypotheses.* New York: Wiley, 1959.

[12] L.E. Brennan and I.S. Reed, "A Recursive Method of Computing the Q Function'" *IEEE Trans. on Inform. Theory*, Vol. IT-11, pp.312-313, April 1965.



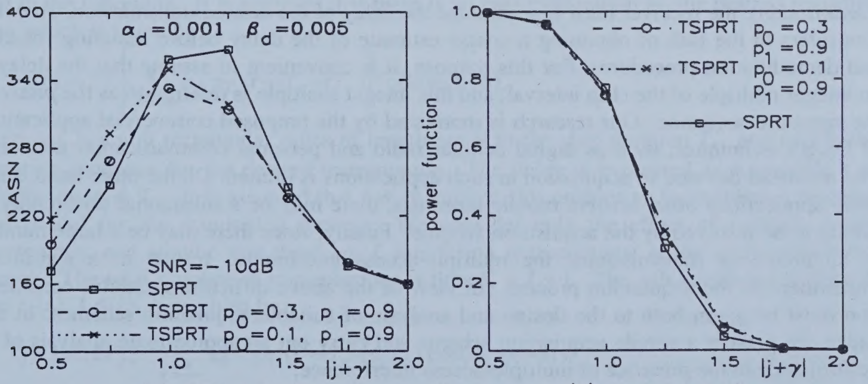
(a) ASN function (b) power function

Figure 2 ASN and power functions for FSS test, SPRT, and TSPRT



(a) ASN function (b) power function

Figure 3 ASN and power functions for FSS test, SPRT, and TSPRT



(a) ASN function (b) power function

Figure 4 ASN and power functions for SPRT and TSPRT

ON THE PERFORMANCE OF A TWO-STAGE ACQUISITION SCHEME FOR DIRECT-SEQUENCE SPREAD SPECTRUM

Upamanyu Madhow
Bell Communications Research
445 South Street, Room 2M-347
Morristown, NJ 07960

ABSTRACT

The problem of acquiring a direct-sequence spread-spectrum (DS/SS) signal in the presence of multiple-access interference is considered. The considerations governing the design of the acquisition scheme are as follows. While the timing uncertainty may be small due to tight scheduling of transmissions and a small network radius, there may be a large number of simultaneous transmissions, leading to significant multiple-access interference. For many applications, the requirements on acquisition time make serial search schemes unacceptable. A passive matched filter scheme is also infeasible in view of the limitations (based on cost and technology) on the length of the filter. In view of these considerations, we propose and analyze a hybrid scheme which employs a programmable matched filter for initial detection, followed by a correlator for verification. Our results demonstrate that the problem of acquisition may limit the size and performance of DS/SS networks. Further research is necessary, therefore, both for the design and performance evaluation of suitable acquisition schemes.

I. INTRODUCTION

We consider the problem of acquiring a direct-sequence spread-spectrum (DS/SS) signal in the presence of multiple-access interference. A DS/SS signal is generated by spreading the data signal using a spreading, or signature, sequence. The signature sequence consists of a sequence of pulses, or chips, of a duration much smaller than the duration of a data symbol. Several DS/SS signals can be reliably transmitted over the same channel by an appropriate choice of signature sequences with good crosscorrelation properties. However, in order to despread and demodulate a given transmission of interest (henceforth called the *desired transmission*), the receiver must first estimate the delay of the desired transmission. *Acquisition* refers to the task of obtaining a coarse estimate of the delay before initiating tracking and demodulation procedures. For this purpose, it is convenient to assume that the delay is an integer multiple of the chip interval, and this integer multiple is referred to as the *phase* of the signature sequence. Our research is motivated by the proposed commercial applications of DS/SS techniques, such as digital cellular radio and personal communication networks. The overhead devoted to acquisition in such applications is limited. On the other hand, since these applications often involve mobile terminals, there may be a substantial uncertainty in timing to be resolved by the acquisition scheme. Finally, since there may be a large number of simultaneous transmissions, the multiple-access interference results in a significant impairment to the acquisition process. In view of the above difficulties, careful consideration must be given both to the design and analysis of suitable acquisition schemes. In this paper, we present a simple acquisition scheme and carry out an approximate analysis of its performance in the presence of multiple-access interference.

Acquisition in the presence of additive white Gaussian noise has been extensively investigated in the literature (see [2], [8], and the references therein), but the effect of multiple-access interference on the acquisition process has not been adequately explored. The effect of multiple-access interference on a passive matched filter acquisition scheme is considered in [5] [6], and an asymptotic analysis as the matched filter length increases shows that the problem of acquisition may limit the maximum number of simultaneous transmissions more severely than post-acquisition criteria such as the bit error probability. However, a passive matched filter scheme is infeasible for systems with a large number of simultaneous transmissions, since the lengths of the matched filters available are limited by cost and technology. Thus, initial detection using a matched filter must be verified by other means. Another alternative is not to use a matched filter at all, but to employ an active correlator that searches serially through the possible phases of the signature sequence of the desired transmission. However, the length of the correlation period required for checking each phase reliably increases with the number of simultaneous transmissions, so that the transmission overhead for resolving even a small timing uncertainty can be prohibitively large.

The above considerations motivate us to consider a hybrid scheme that consists of a programmable matched filter for initial detection, followed by a correlator for verification. The design is aimed at resolving a small timing uncertainty in the presence of significant multiple-access interference, and is subject to a constraint on the length of the matched filter used. Detailed description of the scheme is postponed to the next section. However, it is worth noting that since the initial timing uncertainty is small, the scheme is designed to ensure that throughout the acquisition process, the timing uncertainty never gets larger than the initial value. We will see that this consideration makes it necessary for the matched filter to be programmable, and also makes the mathematical model for the acquisition process somewhat different from that for similar two-stage schemes considered in the literature [8] [2, p. 419]. The performance criterion of interest is the probability of acquiring the desired transmission within a given time $T_{\max}$. This criterion is consistent with our assumption that the acquisition time is critical, and is more appropriate than standard criteria such as the expected acquisition time.

The system model is given in Section II. In Section III, we model the acquisition process as a renewal process, and analyze it on that basis. Section IV contains some numerical results, and Section V gives our conclusions.

II. SYSTEM MODEL

The model for the desired and interfering transmissions is as in [5] [6], and is summarized in the following. It is assumed that there is no data modulation on the desired transmission, which is given by

$$r_0(t) = 2^{1/2} \left[\sum_{j=-\infty}^{\infty} a_j \, p_{T_c}(t - jT_c - \tau T_c) \right] \cos(\omega_c t + \phi),$$

where p_{T_c} is a unit rectangular pulse of length equal to the chip duration T_c, and (a_j) is the signature sequence for the desired transmission. The delay is expressed as a multiple τ of the chip interval T_c. It is assumed that τ is an integer; this enables us to establish a discrete-time model for the acquisition process. We consider only the effect of multiple-access interference, and assume that there are J interfering transmissions at the same carrier frequency. The total number of transmissions is thus $K = J + 1$. The j-th interfering transmission $r_j(t)$, $1 \le j \le J$, is given by

$$r_j(t) = (2P_j)^{1/2} \sum_{k=-\infty}^{\infty} x_k^{(j)} \, p_{T_c}(t - kT_c - \tau_j T_c) \cos(\omega_c + \phi + \theta_j),$$

where P_j is the power relative to that of the desired transmission and θ_j is the carrier phase relative to that of the desired transmission. The sequence $(x_k^{(j)})$ results from multiplying the data sequence (if present) and the signature sequence of the interfering transmission, and the delay τ_j is expressed modulo the chip interval, taking its value in the interval [0,1]. The additive interference is thus given by $X(t) = \sum_{j=1}^{J} r_j(t)$, and the net received signal is $r(t) = r_0(t) + X(t)$.

The sequences (a_k) and $(x_k^{(j)})$ are modeled as random and independent. For each of these sequences, the elements are assumed to be independent and identically distributed, taking values +1 or -1 with equal probability. This model is referred to here as the *random sequence model.* In practice, the signature sequence for each transmission is deterministic and periodic, although data modulation, if present, may be modeled as random. However, the random sequence model is particularly useful in obtaining rough performance estimates for large systems whose performance is difficult to characterize exactly either through analysis or simulation. The model has been used previously not only in bit error probability evaluations [4], [7], but also in modeling a different aspect of the acquisition problem [1].

It is assumed that the receiver can acquire the carrier frequency and the phase of the target transmission perfectly. The receiver computes the statistics

$$Z_k = \frac{2^{1/2}}{T_c} \int_{kT_c}^{(k+1)T_c} r(t) \cos(\omega_c t + \phi)\, dt = a_{k-\tau} + X_k, \tag{1}$$

where the additive interference X_k is given by

$$X_k = \sum_{j=1}^{J} P_j^{1/2} \cos\theta_j \, [(1-\tau_j) x_k^{(j)} + \tau_j x_{k-1}^{(j)}]. \tag{2}$$

The index of the element of the desired signature sequence contributing to the sequence (Z_k) at a given time is called the *current phase* of the signature sequence. From (1), the current phase at time k is $k - \tau$.

For simplicity and concreteness, we assume that all transmissions have equal power ($P_j = 1$, $1 \le j \le J$ in (2)); the analysis is essentially unchanged for unequal powers. There are two models commonly considered in the literature for the relative carrier phases θ_j and the relative delays τ_j of the interfering transmissions. The first is a *synchronous* multiple-access system, in which the interfering transmissions are synchronized to the desired transmission in both carrier phase and chip timing, so that $\theta_j = 0$ and $\tau_j = 0$. The second is an *asynchronous* multiple-access system, in which the θ_j and τ_j are modeled as independent random variables: the θ_j are uniformly distributed over $[0,2\pi]$ and the τ_j are uniformly distributed over [0,1]. We also consider an *asynchronous baseband* system for modeling the multiple-access interference resulting from signaling using M-ary phase shift keying ($M > 2$); here the delays τ_j are modeled as for the asynchronous system, and the carrier phases θ_j are set to zero as in the synchronous system. Among all possible models of the delays and phases of the interfering transmissions, the synchronous system, which is easier to analyze, is known to have the worst-case performance from the point of view of bit error probability [9]. Hence, consideration of the synchronous system is expected to yield pessimistic results for the acquisition problem as well. The asynchronous and asynchronous baseband systems are perhaps more accurate models for practical systems, but are more difficult to analyze due to the dependencies introduced by the asynchronism. Most of these dependencies can be eliminated by conditioning the acquisition process on the relative carrier phases θ_j and the relative delays τ_j appearing in (2), computing the performance measure of interest, and subsequently removing the conditioning. Such methods have been used previously in bit and packet error probability computations [4] [7].

Our task is to estimate τ based on the sequence of statistics (Z_k). We assume that the acquisition process starts at time zero, and that τ is known to take an integer value in the interval $[1,T]$, where T is an integer. We are interested in the situation in which the initial timing uncertainty T is small by virtue of tight control of transmission times, and hence our acquisition scheme is designed to satisfy the property that the timing uncertainty never exceeds its initial value of T. In the following, we describe the acquisition scheme.

The Z_k are the input to a discrete-time filter of length N. At any given time in the process of acquisition, the filter is matched to a section $a_{r-N+1}, a_{r-N+2}, \ldots, a_r$ of the signature sequence of the desired transmission; that is, the filter coefficients are given by $h_i = a_{r-i}$, $i = 0,1,\ldots,N-1$. The integer r is called the *targeted phase*, and may be changed during the process of acquisition by reprogramming the filter. The matched filter output (W_n) is given by

$$W_n = \sum_{i=0}^{N-1} h_i \, Z_{n-i} \, .$$

We describe the process starting from time r; the targeted phase is r, and the current phase is $r - \tau$. Note that this condition is satisfied for the initial value of $r = 0$. The matched filter output is monitored at intervals of unit length. The desired transmission contributes a peak of height N to the matched filter output at time $r + \tau$ (when the current phase equals the targeted phase). Since $1 \le \tau \le T$, we expect a peak by time $r + T$, and try to detect it by means of a threshold rule. Specifically, for $\alpha \in [0,1]$, a *hit* is said to occur when the matched filter output W_n exceeds the threshold αN for some $r + 1 \le n \le r + T$. If there is no hit by time $r + T$, we know that we have failed to detect the peak at time $r + \tau$ (since $\tau \in [0,T]$). We reprogram the matched filter coefficients to accomodate the resulting delay, replacing r by $r + T$, and continue monitoring the matched filter output. This preserves the initial timing uncertainty.

For $t \ne r + \tau$, a hit at time t is called a *false alarm*. A hit at time $t = r + \tau$ is called a *correct hit*, and the absence of a hit at that time is called a *miss*. If a hit does occur, we stop monitoring the matched filter output and start verifying the hit by correlating the matched filter input with the appropriate section of the desired signature sequence over an interval C. Thus, if there is a hit at time t; that is, if

$$t = \min \{ r + 1 \le n \le r + T : W_n / N > \alpha \},$$

then we correlate Z_k, $t + 1 \le k \le t + C$ with $a_{r+1}, \ldots, a_{r+C}$ to obtain the verification statistic

$$V = \sum_{i=1}^{C} a_{r+i} Z_{t+i} \, .$$

The verification statistic is compared to a threshold to determine whether a hit is authenticated. For $\beta \in [0,1]$, if $V > \beta C$, then the hit is authenticated, and the acquisition process terminates (typically, tracking and demodulation are initiated at this point). If the hit is correct, then we have *successful acquisition* upon authentication. On the other hand, if the hit is a false alarm and is authenticated, then we have an *overall false alarm*. We assume that the opportunity to acquire the desired signal is lost by the time the receiver recovers from the tracking and demodulation procedures initiated by an overall false alarm. This is typical of applications in which the acquisition time is critical. If $V \le \beta C$, the hit is rejected, and we resume monitoring the output of the matched filter. However, in order to account for the delay in verification, the filter coefficients are matched to a later section of the desired signature sequence. Specifically, we set $r = t + C$, so that a peak is expected again after a time τ (when the current phase equals r), thus preserving the initial timing uncertainty.

In both the situations under which the matched filter is reprogrammed, we revert to the conditions at the beginning of the acquisition process by preserving the relative delay $\tau \in [0,T]$ between the targeted phase and the current phase of the desired signature sequence. If neither of these two situations occur, the acquisition process terminates in either successful acquisition or overall false alarm. The foregoing description suggests viewing the acquisition process as a renewal process with absorbing states, and our analysis is based on this view. We note that a renewal process model is an approximation even for the random sequences considered in this paper. This is because the portion of the acquisition process that starts after reprogramming the matched filter is not independent of the past, since some elements of the input sequence (Z_k) contribute both to past decisions and to the matched filter output for the new process. However, the dependence is weak, and is ignored in our analysis.

III. MATHEMATICAL MODEL AND ANALYSIS

For the rest of this paper, we make the worst-case assumption $\tau = T$. We first describe the renewal process model for the acquisition process, and then give approximate formulas based on the central limit theorem for the transition probabilities for the renewal model. Although it is easy to develop a recursive formula for the acquisition probability P_{SA} based on the renewal model, a very good approximation $\tilde{P}_{SA}$ for the acquisition probability that is much easier to compute can also be obtained. Due to space limitations, we omit the recursive formula, and present the approximate formula and upper and lower bounds for the acquisition probability. However, the numerical results in Section IV include the exact acquisition probability as well.

A. Development of the renewal process model

It is assumed that there is a renewal whenever the matched filter is reprogrammed. The possible states of the acquisition process are described in the following. State i corresponds to the current phase being a_{r+i-T} (since we assume $\tau = T$), where $1 \le i \le T$. There are two absorbing states, ACQ and FA, corresponding to successful acquisition and overall false alarm, respectively. In addition, there are two intermediate states, VER_f and VER_c. The process starts from state 1, and a renewal consists of a return to state 1. For $1 \le i \le T-1$, a transition from state i can lead either to VER_f or to $i+1$; a transition to VER_f corresponds to initiation of the verification stage following a false alarm, and a transition to $i+1$ occurs if there is no false alarm. Rejection of the false alarm leads to a renewal, and authentication to absorption into FA. From the state $\tau = T$ corresponding to the targeted phase, there is a renewal if there is a miss, otherwise we proceed to VER_c, which corresponds to initiation of the verification stage following a correct hit. From VER_c, authentication of the hit leads to absorption into ACQ, and rejection leads to a renewal. In general, all transitions from VER_f and VER_c take time C, and all other transitions take unit time.

Note that, if the process is in state i, it must be true that there have been no false alarms in the previous $i-1$ states that have been visited since the last renewal. Thus, transition probabilities out of state i must be computed by conditioning on the latter event. Since the interference is likely to be less harmful subject to such conditioning, we argue that replacing the conditional probability of a false alarm at state i by the unconditional probability of false alarm leads to a pessimistic approximation to the acquisition probability. Similar approximations are made for other transition probabilities as well.

We also make an approximation regarding the contribution of the target transmission to the matched filter output. When the current phase equals the targeted phase, the desired transmission contributes a peak of height N to the matched filter output. At other instants, we model its contribution to the matched filter output as resulting from an independent interfering transmission; this, together with the J original interfering transmissions, constitutes the *net interference*. Let Y_n be the matched filter output at time n due to this net interference, and let W_n be the net output at time n. Thus, after each renewal (with targeted phase r), we have that $W_n = Y_n$, $n \ne r+\tau$, and $W_{r+\tau} = Y_{r+\tau} + N$. A similar approximation is made regarding the correlator output V. When a correct hit is being verified, the correlator output $V = C + V'$, and when a false alarm is being verified, $V = V'$, where V' is the contribution of the net interference to the output of the correlator.

Using the above approximations, we obtain the signal flow graph representation of the acquisition process shown in Figure 1. A transition labeled pz^l has probability p and takes time l. The transition probabilities shown are specified as

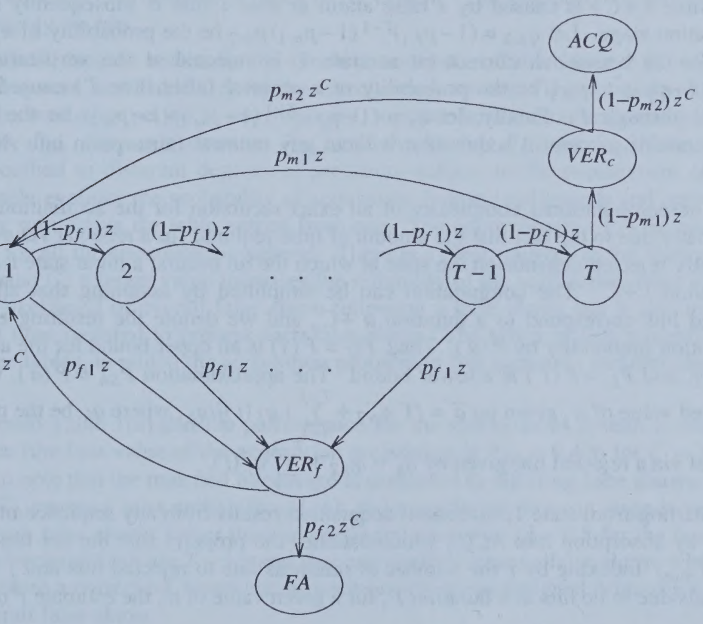
$$p_{f1} = P[Y_1 > \alpha N] = P[Y_i > \alpha N], \quad 1 \le i \le T-1, \tag{3}$$

$$p_{m1} = P[Y_T \le (1-\alpha)N], \tag{4}$$

$$p_{f2} = p(VER_f, FA) = P[V' > \beta C], \tag{5}$$

$$p_{m2} = p(VER_c, 1) = P[V' \le -(1-\beta)C]. \tag{6}$$

Figure 1 Signal flow graph representation of acquisition process



C. Computation of transition probabilities

Since we consider large values of N, C and J, it is possible to approximate the transition probabilities (3)-(6) accurately using the central limit theorem. These approximations are employed in our numerical results, and are given in the following for all three systems under consideration: synchronous, asynchronous, and asynchronous baseband. No derivation is given, but applications of the central limit theorem in similar situations are found in [4]-[7], [9].

Let $Q(\cdot)$ be the complementary standard Gaussian distribution function. For the synchronous system, we have

$$p_{f\,1}(sync) \approx Q\,[\alpha[N/(J+1)]^{1/2}], \quad p_{m\,1}(sync) \approx Q\,[(1-\alpha)(N/J)^{1/2}], \tag{7}$$

$$p_{f\,2}(sync) \approx Q\,[\beta[C/(J+1)]^{1/2}], \quad p_{m\,2}(sync) \approx Q\,[(1-\beta)(C/J)^{1/2}]. \tag{8}$$

For the asynchronous system, we obtain

$$p_{f\,1}(async) \approx Q\,[\alpha N^{1/2}/(J/3+1)^{1/2}], \quad p_{m\,1}(async) \approx Q\,[(1-\alpha)N^{1/2}/(J/3)^{1/2}], \tag{9}$$

$$p_{f\,2}(async) \approx Q\,[\beta C^{1/2}/(J/3+1)^{1/2}], \quad p_{m\,2}(async) \approx Q\,[(1-\beta)C^{1/2}/(J/3)^{1/2}]. \tag{10}$$

For the asynchronous baseband system, the formulas are given by

$$p_{f\,1}(async\,,bb) \approx Q\,[\alpha N^{1/2}/(2J/3+1)^{1/2}], \quad p_{m\,1}(async\,,bb) \approx Q\,[(1-\alpha)N^{1/2}/(2J/3)^{1/2}], \tag{11}$$

$$p_{f\,2}(async\,,bb) \approx Q\,[\beta C^{1/2}/(2J/3+1)^{1/2}], \quad p_{m\,2}(async\,,bb) \approx Q\,[(1-\beta)C^{1/2}/(2J/3)^{1/2}]. \tag{12}$$

C. Evaluation of the acquisition probability

For $1 \le i \le T-1$, let $q_f(i) = (1-p_{f\,1})^{i-1} p_{f\,1}(1-p_{f\,2})$ be the probability that a renewal (after time $i+C$) is caused by a false alarm at state i that is subsequently rejected at the verification stage. Let $q_{m\,2} = (1-p_{f\,1})^{T-1}(1-p_{m\,1})p_{m\,2}$ be the probability of a renewal (after time $T+C$) because a correct hit at state T is rejected at the verification stage. Let $q_{nh} = (1-p_{f\,1})^{T-1}p_{m\,1}$ be the probability of a renewal (after time T) caused by no hits at states 1 through T. Finally, let $q_s = (1-p_{f\,1})^{T-1}(1-p_{m\,1})(1-p_{m\,2})$ be the probability of occurrence of successful acquisition without any renewal (absorption into ACQ after time $T+C$).

The computational complexity of an exact recursion for the acquisition probability is principally due to the fact that the amount of time required for a renewal via a hit that is subsequently rejected depends on the state at which the hit occurs; a hit at state i corresponds to a duration $i+C$. The computation can be simplified by assuming that all renewals via rejected hits correspond to a duration $a+C$, and we denote the resulting estimate of the acquisition probability by $\tilde{P}(a)$. Thus, $P_U = \tilde{P}(1)$ is an upper bound for the acquisition probability, and $P_L = \tilde{P}(T)$ is a lower bound. The approximation $\tilde{P}_{SA} = \tilde{P}(\bar{a})$, where $\bar{a}$ is the expected value of a, given by $\bar{a} = \{T\,q_{m\,2} + \sum_{i=1}^{T-1} i\,q_f(i)\}/q_h$, where q_h be the probability of a renewal via a rejected hit, given by $q_h = q_{m\,2} + \sum_{i=1}^{T-1} q_f(i)$.

Starting from state 1, successful acquisition results from any sequence of renewals, followed by absorption into ACQ, which satisfies the property that the net time elapsed is at most $T_{\max}$. Indexing by i the number of renewals due to rejected hits and j the number of renewals due to no hits in a duration T, for a given value of a, the estimate $\tilde{P}(a)$ is given by

$$\tilde{P}(a) = q_s \sum_{(i,j)\in A} C(i+j,j)\,q_h^{\,i}\,q_{nh}^{\,j},$$

where the set $A = \{(i,j) |\, i(a+C) + jT + (T+C) \le T_{\max}\}$ depends on a, and $C(n,m) = n!/[m!\,(n-m)!]$.

IV. NUMERICAL RESULTS

The numerical evaluation is done for each of the three systems of interest: synchronous, asynchronous, and asynchronous baseband. We consider some typical values for the system parameters. The timing uncertainty T and the number $K = J+1$ of transmissions are fixed by the network size and the transmission protocol. The matched filter length is fixed to the maximum possible length as determined by considerations of cost and technology. The time $T_{\max}$ available for acquisition is determined by the transmission overhead allowed. For a fixed set of values of these parameters, and for several different values of the correlation period C, we compute the performance for the best values of the thresholds α and β. The latter values are determined by a search based on numerical evaluation of the performance. The search for best α and β for a given C employs the approximation $\tilde{P}_{SA}$ defined in Section III.C. However, the approximate acquisition probability corresponding to the optimal values thus obtained is compared with the exact value P_{SA} obtained using a recursion that is omitted from this paper. The agreement is excellent, and indicates that the approximation $\tilde{P}_{SA}$, which is very simple to evaluate, is adequate for use in optimizing the system parameters. We also compute the upper bound P_U and the lower bound P_L (specified in Section III.C) for the acquisition probability for the optimal values of the threshold.

We take the total number of transmissions K to be 50, and the timing uncertainty T to be 30 chips. The matched filter length is taken to be $N = 100$, and the time $T_{\max}$ available for acquisition is taken to be 6400 chips. These numbers are loosely based on typical specifications for cellular spread-spectrum systems proposed in the literature. The chip rate is half the spread bandwidth, which is roughly 12 MHz. The date rate is approximately 1 Kb/s, and the radius of the network is taken to be 1-2 km. The overhead for acquisition is taken to be roughly ten bits.

Table 1(a) displays the results for the synchronous system, Table 1(b) for the asynchronous baseband system, and Table 1(c) for the asynchronous system. The optimal values of α and β for each correlation period C are displayed, together with the corresponding acquisition probability P_{SA}, the approximation $\tilde{P}_{SA}$, and the bounds P_U and P_L. Note that the entries are specified to different degrees of precision, subject to the requirement of being able to accurately evaluate the probability of acquisition failure (and bounds and approximations for it) by subtracting the table entries from one. The approximation $\tilde{P}_{SA}$ is seen to be very accurate for all the situations considered. There is a clear tradeoff in the choice of the correlation period C: a larger value increases the reliability of the verification stage and thus decreases the likelihood of overall false alarm following a false alarm, but it also increases the amount of time required to process each false alarm. The effect of this tradeoff is evident in all three tables, in which the acquisition probability first increases and then decreases with C.

We see from Table 1(a) that the performance for the synchronous system is unacceptable in practice (the best value of the acquisition probability is $P_{SA} = 0.406$ for $C = 300$). It is interesting to note that the matched filter stage is dedicated to rejecting false alarms ($\alpha = 1$, which, from (7), causes a miss probability of ½), and even the verification stage is weighted towards rejecting false alarms rather than processing a correct hit ($\beta = 0.8$ for the best value of C). This is because there are $T - 1$ states that can possibly cause a false alarm, while only one state can cause a correct hit, hence the acquisition scheme is designed primarily to guard against an overall false alarm.

The performance for the asynchronous baseband system (see Table 1(b)) is far better than in the synchronous system, although it may still not be acceptable for a system design (the best value of acquisition probability is $P_{SA} = 0.88$ for $C = 300$). The performance for the asynchronous system (see Table 1(c)) is much superior to both the preceding systems, and is definitely acceptable for any system design (the best value of acquisition probability is $P_{SA} = 0.99995$ for $C = 400$). As in Table 1(a), the optimal values of α and β in Tables 1(b) and 1(c) are high as well, guarding against false alarm rather than miss.

Table 1 Optimal thresholds and acquisition probability

(a) Synchronous system

C	α	β	P_{SA}	$\tilde{P}_{SA}$	P_U	P_L
100	1	1	.245	.245	.245	.244
200	1	1	.399	.400	.406	.381
300	1	.8	.406	.406	.414	.395
500	1	.7	.366	.366	.366	.364
800	1	.6	.275	.275	.275	.275

(b) Asynchronous baseband system

C	α	β	P_{SA}	$\tilde{P}_{SA}$	P_U	P_L
100	1	1	.700	.700	.701	.699
200	1	1	.867	.867	.876	.855
300	1	.8	.880	.881	.885	.872
500	1	.7	.839	.839	.839	.825
800	1	.6	.707	.707	.707	.707

(c) Asynchronous system

C	α	β	P_{SA}	$\tilde{P}_{SA}$	P_U	P_L
200	1	1	.9996	.9996	.9996	.9996
300	1	.9	.9999	.9999	.9999	.9999
400	1	.8	.99995	.99995	.99995	.99994
800	1	.6	.9993	.9993	.9993	.9993
1000	1	.6	.9981	.9981	.9982	.9979

V. CONCLUSIONS

Our numerical results show that an appropriate choice of α, β and C are critical to the performance. These depend on the other system parameters, such as the timing uncertainty, the number of transmissions, and the matched filter length. Asynchronism appears to result in tremendous performance gains, and that is encouraging, since the asynchronous and asynchronous baseband systems are probably more accurate models for practical systems than the synchronous system. It is clear, however, that the performance even for an asynchronous baseband system may not be adequate for typical applications. Further, in many applications of interest (such as mobile radio), there are further impairments to performance due to fading and multipath. Thus, further research is needed for devising and evaluating good acquisition schemes.

For the systems considered here, one method for improving performance is simply to alter the system parameters: increase the matched filter length (this depends on cost and technology issues), decrease the timing uncertainty (this depends on network protocol issues), or increase the available acquisition time (this depends on the available overhead for acquisition). We must also consider other acquisition schemes for the same system, some of which may be obtained through simple modifications of the scheme presented here; for instance, we can employ sequential analysis techniques [1] to reduce the average verification period. The final approach is to consider systems with more information than is assumed available in this paper, such as systems that employ a pilot tone for synchronization purposes. Our results indicate that all these options merit careful consideration in designing DS/SSMA systems which support a large number of users.

ACKNOWLEDGEMENTS

This research was sponsored in part by the Joint Services Electronics Program under contract N00014-90-J-1270 and in part by the Illinois Department of Commerce and Community Affairs under grant 90-82141.

REFERENCES

[1] K. K. Chawla and D. V. Sarwate, "Coherent acquisition of PN sequences using a random sequence model and the SPRT," *Proc. Ann. Conf. on Info. Sci. and Systems,* Princeton University, Princeton, NJ, pp. 52-57, March 1990.

[2] J. K. Holmes, *Coherent Spread Spectrum Systems.* New York: Wiley, 1982.

[3] W. Feller, *An Introduction to Probability Theory and Its Applications,* vol. 2. New York: Wiley, 1971.

[4] J. S. Lehnert and M. B. Pursley, "Error probabilities for binary direct-sequence spread-spectrum communications with random signature sequences," *IEEE Trans. Commun.,* vol. COM-35, no. 1, pp. 85-96, January, 1987.

[5] U. Madhow and M. B. Pursley, "Acquisition-based capacity of direct-sequence spread-spectrum communication networks," *Proc. Ann. Conf. on Info. Sci. and Systems,* The Johns Hopkins University, Baltimore, MD, pp. 518-522, March 1991.

[6] U. Madhow and M. B. Pursley, "Acquisition in direct-sequence spread-spectrum communication networks: an asymptotic analysis," to be submitted for publication.

[7] R. K. Morrow, Jr. and J. S. Lehnert, "Bit-to-bit dependence in DS/SSMA packet systems with random signature sequences," *IEEE Trans. Commun.,* vol. COM-37, no. 1, pp. 1052-1061, October, 1989.

[8] A. Polydoros and C. L. Weber," A unified approach to serial search spread-spectrum code acquisition--Part II: a matched filter receiver," *IEEE Trans. Commun.,* vol. COM-32, no. 5, pp. 550-560, May 1984.

[9] M. B. Pursley, "Performance evaluation for phase-coded spread-spectrum multiple-access communication-Part I: System analysis," *IEEE Trans. Commun.,* vol. COM-25, no. 8, pp. 795-799, August 1977.

Optimum Multiuser Detection in Single-Path Rayleigh Fading Channels

Zoran Zvonar, David Brady
Department of Electrical and Computer Engineering
Northeastern University
Boston, MA 02115

Abstract

In this paper the performance of the optimum multiuser detector in the single-path Rayleigh fading channel is considered. The received signal is additively composed of Gaussian noise and the signal of K asynchronous users, each passed through an independent Rayleigh fading channel. Assuming perfect channel estimates, the optimum multiuser detector consists of a bank of single user matched filters followed by the dynamic programming algorithm of the Viterbi type. Bounds on the single user error rate are presented and compared to those for the conventional receiver and isolated transmission and the notion of asymptotic multiuser efficiency is extended for this channel. Previous work in the area indicates that the conventional receiver suffers from the near-far problem imposed by the fading nature of the channel. In this paper it is shown that in the high average signal-to-noise ratio region, the performance of the optimum receiver rivals that of isolated transmission and indicates that the near-far problem can be alleviated by more sophisticated receiver design.

1 Introduction and Channel Model

Previous work in the area of the multiuser detection has concentrated on the additive white Gaussian noise (AWGN) channel. It includes the work on the optimum multiuser detector which was proven to be near-far resistant [1], indicating that the near-far problem is not an inherent shortcoming of the DS/SSMA systems but of conventional detection. On the other side, the complexity of the optimum multiuser detector is exponential in the number of users. Suboptimal, near-far resistant detectors were subsequently developed which have significantly reduced complexity, including the multiple-access counterparts to the zero-forcing equalizer [2], the decision-directed equalizer [3] and sequential decoder [4].

Several important multiple-access channels exhibit severe multipath fading, which limits system performance [5]. Examples include cellular radio, indoor wireless communications, personal communications and underwater LAN. The purpose of this paper is to address the performance of multiuser detction in the single-path Rayleigh fading channel, which is an introduction to the more complex, multipath fading environment. Particularly, the restatement of the near-far problem created by Rayleigh fading channels is investigated. We show that optimal detection rivals isolated transmission, and conclude that the performance limits noted in [6] are not inherent to the multiple-access fading channel but results from the suboptimal detector.

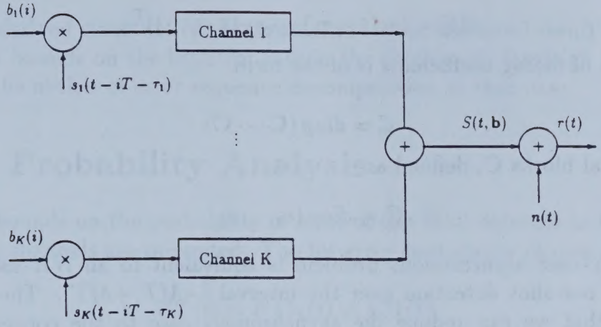


Figure 1: Single-path fading multiple-access channel model

Consider the coherent asynchronous multiple-access system depicted in Figure 1. The k^{th} user employs binary modulation (BPSK) to transmit $N = 2M + 1$ data bits. $b_k(i) \in \{\pm 1\}$, $-M \leq i \leq M$, at a common symbol rate $1/T$. The aggregate equivalent baseband signal at the multiuser receiver is given by

$$r(t) = S(t, \mathbf{b}) + n(t) \tag{1}$$

where n(t) is a complex, Gaussian noise with power spectral density N_0 and zero mean, and

$$S(t, \mathbf{b}) = \sum_{i=-M}^{M} \sum_{k=1}^{K} c_k \, b_k(i) \, s_k(t - iT - \tau_k) \tag{2}$$

where $c_k s_k(t)$ is the received baseband signal for user k. Without loss of generality the users are ordered according to their relative delays $0 < \tau_1 < \tau_2 < \ldots < \tau_K < T$. The signal of each user arrives at the receiver through an independent, single-path channel. Since each channel is assumed to be Rayleigh faded, the multiplicative coefficients c_k are independent, zero-mean, complex-valued Gaussian random variables, with variances $\overline{| c_k |^2}$ and independent quadrature components. Except for the fading coefficients , the model is the same as one introduced in [1].

To analyze the asynchronous transmission case, an equivalent one-shot synchronous problem [2] is considered. Using compact matrix notation the received signal is given by

$$r(t) = \mathbf{b}^T \mathcal{C} \mathbf{s}_t + n(t) \tag{3}$$

where the equivalent data vector is from $N \times K$ users

$$\mathbf{b}^T = (\, b_1(-M)\, b_2(-M) \cdots b_K(-M) \cdots b_1(M)\, b_2(M) \cdots b_K(M)\,) \, , \tag{4}$$

and the equivalent signal vector is given by

$$\mathbf{s}_t^T = \left(\mathbf{s}^T(t + MT) \cdots \mathbf{s}^T(t - MT) \right) \tag{5}$$

where

$$\mathbf{s}(t) = (s_1(t-\tau_1)\cdots s_K(t-\tau_K))^T . \tag{6}$$

The matrix of fading coefficients is of the form

$$\mathcal{C} = diag\,(\mathbf{C}\cdots\mathbf{C}) \tag{7}$$

with N diagonal blocks $\mathbf{C}$, defined as

$$\mathbf{C} = diag\,(c_1\cdots c_K)\,. \tag{8}$$

Thus the K-user asynchronous problem is equivalent to an NK-user synchronous problem, with one-shot detection over the interval $[-MT, +MT]$. The significance of this result is that we can reduce the asynchronous case to the conceptually simpler synchronous problem.

2 Multiuser Maximum-Likelihood Sequence Detection

The optimum multiuser receiver uses full knowledge of the signal waveforms $s_k(t)$, fading coefficients c_k, and relative delays τ_k in order to decide for the transmitted sequence $\mathbf{b}$. Since all transmitted sequences are assumed to be equiprobable, this detector selects the information sequence $\mathbf{b}$ which maximizes the log-likelihood function

$$L(\mathbf{b}) = 2Re\left\{\mathbf{b}^H\mathbf{y}\right\} - \mathbf{b}^H\mathcal{C}^H\mathcal{R}\mathcal{C}\mathbf{b} \tag{9}$$

where superscript H denotes conjugate transpose, $\mathbf{y}$ is the sampled vector at the output of the matched filters, and $\mathcal{R}$ is block-tridiagonal, block-Toeplitz crosscorrelation matrix for the equivalent synchronous case

$$\mathcal{R} = \int_{-\infty}^{+\infty} \mathbf{s}_t^*\mathbf{s}_t^T\,dt = \begin{pmatrix} \mathbf{R}(0) & \mathbf{R}(-1) & 0 & \cdots \\ \mathbf{R}(1) & \mathbf{R}(0) & \mathbf{R}(-1) & \cdots \\ & \ddots & \ddots & \\ \cdots & \mathbf{R}(1) & \mathbf{R}(0) & \mathbf{R}(-1) \\ \cdots & 0 & \mathbf{R}(1) & \mathbf{R}(0) \end{pmatrix} \tag{10}$$

with block elements of dimension $K \times K$

$$\mathbf{R}(i-j) = \int_{-\infty}^{+\infty} \mathbf{s}^*(t-iT)\,\mathbf{s}^T(t-jT)dt\,. \tag{11}$$

The sufficient statistics for this decision are the outputs of a bank of matched filters, sampled at the bit epochs of the users

$$\mathbf{y} = \int_{-\infty}^{+\infty} r(t)\mathcal{C}^H\mathbf{s}_t^*\,dt. \tag{12}$$

This detector is called the maximum likelihood sequence (MLS) detector, and may also be interpreted as the receiver which selects the noise realization with the minimum energy. Providing that the fading coefficients are known, the optimal receiver for the single-path fading channel coincides with that for the AWGN channel, which may be implemented with a dynamic programming algorithm of the Viterbi type, and has a demodulation complexity per bit which is exponential in the number of users [1].

Having the derived structure of the optimum receiver the novel result of this work is to provide tight bounds on the bit error rate in the single-path Rayleigh fading channel and to extend the notion of error sequence decomposition in that case.

3 Error Probability Analysis

In this section bounds on the probability of error of the MLS detector in the single-path Rayleigh fading channels are presented. The bit error probability of user k, given as

$$P_k^M(i) = P\left\{b_k(i) \neq \hat{b}_k^M(i)\right\} \tag{13}$$

depends in general on the horizon M and the particular symbol interval i. Since the error rate for bit $b_k(i)$ increases with M, we are interested in the error probability for the infinite horizon case

$$P_k = \lim_{M\to\infty} P_k^M(i)\,. \tag{14}$$

Associated with a given transmitted data sequence $\mathbf{b}$ and maximum likelihood sequence $\widehat{\mathbf{b}}$, there is an error sequence $\mathbf{e} = \frac{1}{2}(\mathbf{b} - \widehat{\mathbf{b}})$. Note that each component of $\mathbf{e}$ is either zero (no error) or is equal to the transmitted symbol. The set of error sequences which produce an error for user k at the i symbol is defined as

$$E_k(i) = \left\{\mathbf{e} \in \{-1,0,1\}^N\,,\, e_k(i) = b_k(i)\right\}. \tag{15}$$

Unfortunately, no closed form expression exists for the probability that the i^{th} bit of the k^{th} user is in error. To upper bound the probability that $\mathbf{e}$ is admissible and maximizes the likelihood function, the usual approach is to use the union bound over $E_k(i)$. For a given realization of the fading coefficients the upper bound may be expressed as

$$P_k^M(i \mid \mathbf{C}) \leq \sum_{\mathbf{e}\in E_k(i)} \frac{1}{2^{w(\mathbf{e})}} Q\left(\frac{\sqrt{2}\|S(\mathbf{e})\|}{\sqrt{N_0}}\right) \tag{16}$$

where the energy of the error sequence is given by

$$\|S(\mathbf{e})\|^2 = \mathbf{e}^H \mathcal{C}^H \mathcal{R} \mathcal{C} \mathbf{e} = \mathbf{c}^H \mathbf{Q} \mathbf{c} \tag{17}$$

with

$$\mathbf{Q} = \sum_{i=-M}^{M} \sum_{j=-M}^{M} \mathbf{E}(i)\,\mathbf{R}(i-j)\,\mathbf{E}(j)\,. \tag{18}$$

and $\mathbf{E}(i) = diag(e_1(i)e_2(i)\cdots e_K(i))$.

Since the channel is random in nature, the bit error probability is obtained by averaging over the fading statistics

$$P_k^M(i) \leq \int_0^\infty \sum_{\mathbf{e}\in E_k(i)} \frac{1}{2^{w(\mathbf{e})}} Q\left(\frac{\sqrt{2x}}{\sqrt{N_0}}\right) p_X(x)dx\,. \tag{19}$$

The matrix $\mathbf{Q}$ is Hermitian and therefore the distribution of $X = \|S(\mathbf{e})\|^2$ is a quadratic form of channel path gains. The characteristic function of X is given by

$$C_X(\omega) = \frac{1}{det(\mathbf{I} - j\omega\mathbf{\Sigma}\mathbf{Q})} \tag{20}$$

where $\mathbf{\Sigma}$ is covariance matrix of the Gaussian random vector $\mathbf{c}$. An alternative expression for the characteristic function is given by

$$C_X(\omega) = \prod_{i=1}^{K} \frac{1}{(1 - j\omega\lambda_i)} = \sum_{i=1}^{K} \frac{A_i}{(1 - j\omega\lambda_i)} \tag{21}$$

where λ_i are eigenvalues of the matrix $\mathbf{\Sigma}\mathbf{Q}$, and the coefficients A_i are the coefficients of a partial fraction expansion.

The average of each term in (16) may be computed exactly using (21), which gives the upper bound on the probability of error

$$P_k^M(i) \leq \sum_{\mathbf{e} \in E_k(i)} 2^{-w(\mathbf{e})} \sum_{i=1}^{K} \frac{A_i}{2} \left[1 - \frac{1}{\sqrt{1 + \frac{N_0}{\lambda_i}}} \right]. \tag{22}$$

Unfortunately, this upper bound diverges due to the great degree of overlap in the union bound. As in the analysis of the MLS detector in the AWGN channel, we are motivated to reduce the amount of overlap by reducing the number of error sequences in the summation. Although concept of indecomposable sequences has provided the tightest known bound for the AWGN channel [1], the set of indecomposable sequences varies according to the random fading coefficients $\mathbf{C}$ and is not directly helpful to the fading channel. For this reason we extend the notion of the decomposability to the fading channel with the following definition.

Definition 1 *An error sequence* $\mathbf{e} \in E_k(i)$ *is* **decomposable** *into* $\mathbf{e}' \in E_k(i)$ *and* $\mathbf{e}'' \in E$ *independently of the channel realization if* $\mathbf{e} = \mathbf{e}' + \mathbf{e}''$, $w(\mathbf{e}') < w(\mathbf{e}), w(\mathbf{e}'') < w(\mathbf{e})$ *and* $Re\left\{\mathbf{c}^H \mathbf{A} \mathbf{c}\right\} \geq 0$ *almost surely, where*

$$\mathbf{A} = \sum_{i=-M}^{M} \sum_{j=-M}^{M} \mathbf{E}'(i)\, \mathbf{R}(i-j)\, \mathbf{E}''(j)\,. \tag{23}$$

Any sequence which is not decomposable will be called *indecomposable* in the sense of this definition, which coincides with the notion of decomposability for the additive Gaussian channel when $\mathbf{C}$ is the identity matrix. The set of indecomposable sequences, $F_k(i)$, as defined in [1] is a subset of the new set of the indecomposable sequences $B_k(i)$ as defined above.

Proposition 1 *For asynchronous transmission over a single-path Rayleigh fading with signal crosscorrelation matrix* $\mathcal{R}$*, the set of indecomposable sequences is the set of error sequences* $B_k(i)$ *with* $e_k(i) \neq 0$ *for some* i *and such that there are no* $K-1$ *or more consecutive zeros amid nonzero components.*

Proof: The third condition of decomposability can be rewritten as

$$Re\left\{ \sum_{i=-M}^{M} \sum_{j=-M}^{M} \mathbf{c}^H \mathbf{E}'^H(i)\, \mathbf{R}(i-j)\, \mathbf{E}''(j) \mathbf{c} \right\} = Re\left\{\zeta^H \mathcal{E}' \mathcal{R} \mathcal{E}'' \zeta\right\} = Re\left\{\zeta^H \mathcal{A} \zeta\right\} \geq 0 \tag{24}$$

where $\mathcal{E}$ is the error diagonal matrix corresponding to the entire error sequence of length NK, and the vector ζ is composed of N blocks of channel vector $\zeta = [\mathbf{c}^T \cdots \mathbf{c}^T]^T$.

Every error sequence $\mathbf{e} \in E_k(i) - B_k(i)$ is decomposable since by taking $\mathbf{e}' \in B_k(i)$ waveforms $S(\mathbf{e}')$ and $S(\mathbf{e}'')$ are orthogonal for any value of fading coefficients.

To establish the equivalence it will be shown that every error sequence $\mathbf{e} \in B_k(i)$ is indecomposable. By the structure of the sequence for any two nonzero elements there is less than $K-1$ zeros between them. Therefore, for any admissible decomposition of an error sequence there must be two integers k and j such that $j - k < K - 1$ and $e'(k)e''(j) \neq 0$. The element of the matrix $\mathcal{A}$ at the position (k,j) is nonzero and element (j,k) is zero or vice versa, depending on the decomposition performed. An exception to this rule exists for particular $\mathcal{R}$, which is not of general interest. Therefore the matrix $\mathcal{B} = \mathcal{A} + \mathcal{A}^H$ is not a zero matrix, but has a zero diagonal and it is not PSD [9], i.e. $\zeta^H \mathcal{B} \zeta < 0$ for some ζ. It is straightforward to verify that $\mathcal{B}$ is not PSD implies $Re\left\{\zeta^H \mathcal{A} \zeta\right\} < 0$ and the third condition for the decomposition can not be fulfilled. QED

The previous proposition provides the smallest set of error sequences in the upper bound invariant to the channel realization. The calculation of the upper bound is performed using the flow graph technique and the one directional stack algorithm introduced in [8].

The lower bound to the performance of the optimum detector is given by a single-user isolated transmission error probability

$$P_k \geq P_k^M(i) \geq \frac{1}{2}\left[1 - \frac{1}{\sqrt{1 + \frac{1}{\gamma_0}}}\right] \tag{25}$$

where

$$\gamma_0 = \frac{E_b}{N_0} \overline{| c_k |^2}$$

which is the case when no other user is active or if the receiver knew the transmitted bits of the other users.

4 Numerical Examples

In this section several numerical examples are presented to illustrate the performance of the conventional detector and MLS multiuser detector, and compare them with the optimum single-user detector for isolated transmission.

Two sets of examples widely adopted in the multiuser literature are considered. They correspond to high and low bandwidth efficiency which are related to the spreading ratio of the DS/SSMA system.

In first example two users are transmitting over the two independent Rayleigh channels with same average path strength. Gold sequences of length J=127 are employed. Using the conventional detector, finite crosscorrelations among the sequences although small in magnitude can cause multiple-access interference limitation due to Rayleigh fading. The saturation level of the error probability for the high SNR is shown to be inversely proportional to the sequence length [6]. The worst case performance of the MLS detector, corresponding to the highest value of the crosscorrelation, is presented in Figure 2. The MLS detector performs close to the optimum detector in isolated transmission and result indicates that the near-far effects may be relieved by appropriate receiver design.

The second example corresponds to a bandwidth efficient case. The signature waveforms of the users are presented in Figure 3. The performance of the MLS detector is given in terms of upper bound for the baseband case. Comparing the results with the previous example, even with the less strict crosscorrelation values, MLS detector performs close to the optimum detector in isolated transmission.

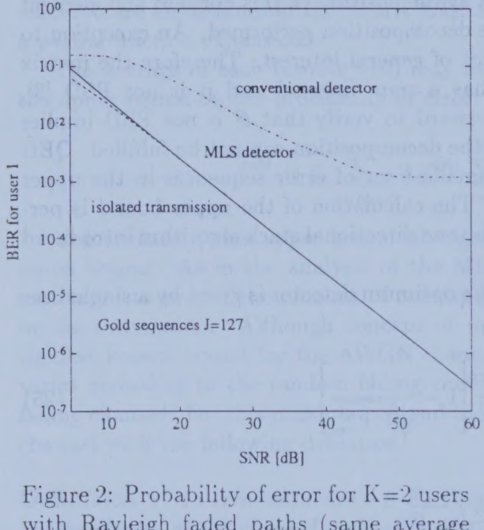


Figure 2: Probability of error for K=2 users with Rayleigh faded paths (same average path strength) using Gold sequences of period 127

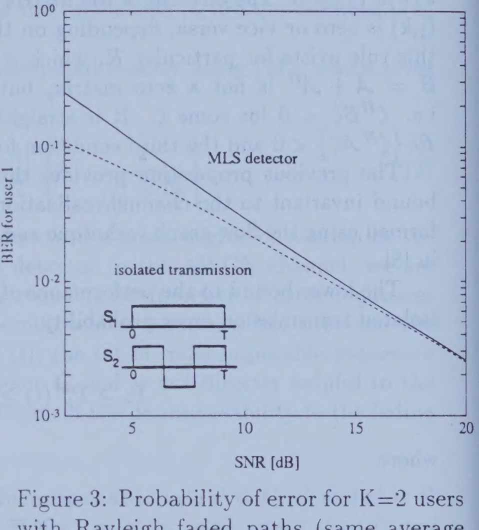


Figure 3: Probability of error for K=2 users with Rayleigh faded paths (same average path strength) , spreading ratio 3

5 Asymptotic Multiuser Efficiency

Probability of error for the isolated transmission in a single-path Rayleigh fading channel for a high SNR is given by

$$P_k^f = \frac{1}{4\gamma_0} + o\left(\frac{1}{\gamma_0}\right) \tag{26}$$

Since the probability of error is a decreasing function of the average SNR, we may describe the performance degradation of the receiver due to MAI through the effective average SNR, γ_{eff}, given as the functional inverse of the probability of error of the particular user. Since the interference is additive and independent of the desired signal, $\gamma_{eff} \leq \gamma_0$. The asymptotic multiuser efficiency is defined for the vanishing noise case following [10].

Definition 2 *For any receiver in a single-path Rayleigh fading channel with a bit error rate P_k^f, the asymptotic multiuser efficiency is given by*

$$\eta_k = \lim_{N_0 \to 0} \frac{\gamma_{eff}}{\gamma_0}$$

where

$$\gamma_{eff} = \frac{1}{4P_k^f} \,.$$

Proposition 2 *For 2 user asynchronous transmission over the single-path Rayleigh fading channel the optimum multiuser detector has the asymptotic multiuser efficiency equal to unity, except when the signals are heavily correlated (normalized crosscorrelation* $\mid \tilde{\rho}_{12} \mid = 1$ *or* $\mid \tilde{\rho}_{21} \mid = 1$*).*

Proof: For any term in the summation for the upper bound, corresponding to a particular error sequence, the eigenvalues of the matrix $\mathbf{\Sigma Q}$ must be calculated. The matrix $\mathbf{Q}$ for the 2 user case is

$$\mathbf{Q} = \begin{pmatrix} w_1 D_{11}(0) & (\tilde{\rho}_{12} D_{12}(0) + \tilde{\rho}_{21} D_{12}(-1))^* \sqrt{w_1 w_2} \\ (\tilde{\rho}_{12} D_{12}(0) + \tilde{\rho}_{21} D_{12}(-1)) \sqrt{w_1 w_2} & w_2 D_{22}(0) \end{pmatrix} \tag{27}$$

where

$$D_{mn}(l) = \sum_{j=-M}^{M} e_m(l+j) e_n(j) \,. \tag{28}$$

The eigenvalues of the matrix $\mathbf{\Sigma Q}$ are given by

$$\begin{aligned} \lambda_p &= \frac{1+\alpha D_{22}(0)}{2} \\ &\quad - \frac{(-1)^p}{2} \sqrt{(1+\alpha D_{22}(0))^2 - 4\alpha D_{11}(0) D_{22}(0) + 4\alpha \mid \tilde{\rho}_{12} D_{12}(0) + \tilde{\rho}_{21} D_{12}(-1) \mid^2} \\ p &= 1,2 \end{aligned} \tag{29}$$

where the ratio of the averaged powers of the users is

$$\alpha = \frac{w_2 \overline{\mid c_2 \mid^2}}{w_1 \overline{\mid c_1 \mid^2}} \,.$$

Equal eigenvalues occur only as a special case which is not of interest. Note that for $\alpha = 0$ the probability of error expression reduces to the single user isolated transmission case. For sufficiently large average SNR the terms in the summation for the upper bound can be expanded in a Taylor series

$$\sum_{i=1}^{2} \frac{A_i}{2} \left[1 - \frac{1}{\sqrt{1+\frac{N_0}{\lambda_i}}} \right] = \begin{cases} \frac{1}{4\gamma_0 \lambda_1} + o\left(\frac{1}{\gamma_0}\right) & \text{for } \lambda_2 = 0 \\ \frac{3}{32\gamma_0^2 \lambda_1 \lambda_2} + o\left(\frac{1}{\gamma_0^2}\right) & \lambda_1 \neq \lambda_2 \neq 0 \end{cases} \tag{30}$$

As long as the eigenvalues are nonzero, the decaying rate of the probability of error of the particular error sequence is larger than for the single-user isolated transmission. The same decaying rate can be obtained only for the case when one of the eigenvalues is equal to zero, which is true for the single error sequence or any other error sequence that satisfies the equality

$$D_{11}(0) D_{22}(0) = \mid \tilde{\rho}_{12} D_{12}(0) + \tilde{\rho}_{21} D_{12}(-1) \mid^2 \,. \tag{31}$$

Denoting the left shifted version of the error sequence $\mathbf{e}_1$ as $\breve{\mathbf{e}}_1$ we can express the condition for the zero eigenvalue in the form

$$\begin{aligned} \langle \mathbf{e}_1, \mathbf{e}_1 \rangle \langle \mathbf{e}_2, \mathbf{e}_2 \rangle &= \mid \langle \widetilde{\rho_{12}} \mathbf{e}_1 + \widetilde{\rho_{21}} \breve{\mathbf{e}}_1, \mathbf{e}_2 \rangle \mid^2 \\ &\leq \langle \mathbf{e}_2, \mathbf{e}_2 \rangle \langle \tilde{\rho}_{12} \mathbf{e}_1 + \tilde{\rho}_{21} \breve{\mathbf{e}}_1, \tilde{\rho}_{12} \mathbf{e}_1 + \tilde{\rho}_{21} \breve{\mathbf{e}}_1 \rangle \\ &= \langle \mathbf{e}_2, \mathbf{e}_2 \rangle \Big(\mid \tilde{\rho}_{12} \mid^2 \langle \mathbf{e}_1, \mathbf{e}_1 \rangle + 2Re \left\{ \tilde{\rho}_{12} \tilde{\rho}_{21}^* \right\} \langle \breve{\mathbf{e}}_1, \mathbf{e}_1 \rangle + \mid \tilde{\rho}_{21} \mid^2 \langle \breve{\mathbf{e}}_1, \breve{\mathbf{e}}_1 \rangle \Big) \end{aligned} \tag{32}$$

Since $\langle \breve{\mathbf{e}}_1, \breve{\mathbf{e}}_1 \rangle = \langle \mathbf{e}_1, \mathbf{e}_1 \rangle$ and $\langle \breve{\mathbf{e}}_1, \mathbf{e}_1 \rangle < \langle \mathbf{e}_1, \mathbf{e}_1 \rangle$ it follows that

$$\langle \mathbf{e}_1, \mathbf{e}_1 \rangle \langle \mathbf{e}_2, \mathbf{e}_2 \rangle \leq \langle \mathbf{e}_2, \mathbf{e}_2 \rangle \Big(\mid \tilde{\rho}_{12} \mid^2 + \mid \tilde{\rho}_{21} \mid^2 + 2 \mid \tilde{\rho}_{12} \mid\mid \tilde{\rho}_{21} \mid \Big) \langle \mathbf{e}_1, \mathbf{e}_1 \rangle \tag{33}$$

with the equality for the case when $(\mid \tilde{\rho}_{12} \mid + \mid \tilde{\rho}_{21} \mid)^2 = 1$. However, $(\mid \tilde{\rho}_{12} \mid + \mid \tilde{\rho}_{21} \mid)^2 \leq 1$, and the equality holds only when the users are heavily crosscorrelated i.e. $\mid \tilde{\rho}_{12} \mid = 1$ or $\mid \tilde{\rho}_{21} \mid = 1$. Following [2] this situation can occur with probability zero if the a priori unknown delays are uniformly distributed .

Since the decay rate of the probability of error of the isolated transmission for high SNR is achieved only by a single error sequence, the probability of error of the MLS detector can be bounded by [1]

$$\frac{1}{4\gamma_0} \leq P_1 \leq \frac{1}{4\gamma_0} + o\left(\frac{1}{\gamma_0}\right) \tag{34}$$

The average effective SNR in that case is given by

$$\gamma_{eff} = \frac{1}{\frac{1}{\gamma_0} + o\left(\frac{1}{\gamma_0}\right)} \tag{35}$$

and for the limiting case of vanishing noise the asymptotic efficiency of the MLS detector is

$$\eta_1 = \lim_{\gamma_0 \to \infty} \frac{1}{1 + \gamma_0 \; o\left(\frac{1}{\gamma_0}\right)} = 1 \tag{36}$$

and the proposition follows. QED

Proposition 3 *The asymptotic multiuser efficiency of the conventional detector in the single-path Rayleigh fading channel is zero.*

Proof: For the purpose of comparison with the conventional detector the results presented in [6] are adopted. The saturation level for the bit error rate for the case of vanishing noise is

$$\lim_{\gamma_0 \to \infty} P_1 \simeq \frac{1}{6N} \tag{37}$$

where N is the period of the applied Gold signature sequences.

From the given expression for the saturation level of bit error rate the asymptotic multiuser efficiency is calculated as

---

[1] $f(x) = o(g(x))$ if $lim_{x \to \infty} \frac{f(x)}{g(x)} = 0$

$$\eta_1 = \lim_{\gamma_0 \to \infty} \frac{3N}{\gamma_0} = 0 \tag{38}$$

QED

References

[1] S.Verdu: " Minimum Probability of Error for Asynchronous Gaussian Multiple Access Channels", IEEE Trans. on Information Theory, Vol IT-32, No 1, January 1986

[2] R.Lupas, S.Verdu: "Near-far Resistance of Multiuser Detectors in Asynchronous Channels", IEEE Trans. Commun. Vol COM-38, April 1990, pp 496-508

[3] M.Varanasi, B.Aazhang: "Multistage Detection in Asynchronous Code-Division Multiple-Access Communications", IEEE Trans. Commun. Vol COM-38, No. 4, April 1990, pp 509-519

[4] Z.Xie, C.Rushforth, R.Short: " Multiuser Signal Detection Using Sequential Decoding", IEEE Trans. Commun. Vol COM-38, No. 5, May 1990, pp 578-583

[5] G.Turin: "The Effects of Multipath and Fading on the Performance of Direct Sequence CDMA Systems" , IEEE JSAC, Vol SAC-2, No 4, July 1984, pp 597-603

[6] M.Kavehrad: " Performance of Nondiversity Receivers for Spread Spectrum in Indoor Wireless Communications", ATT Technical Journal, Vol.64, No.6, July-August 1985, pp 1181-1985

[7] J.Proakis: "Digital Communications", McGraw Hill, 1989

[8] W.Sheen, G.Stuber: "MLSE Equalization and Decoding for Multipath Fading Channels" submitted to IEEE Trans. on Communications

[9] A.Housholder: "The Theory of Matrices in Numerical Analysis", Blaisdell Publishing Company, 1964, pp 12-13

[10] D.Brady: "Asymptotic Multiuser Efficiency for Optical Channels", Proc. 1991 Conference on Information Sciences and Systems, March 1991

Multiuser Detectors for Rician Fading CDMA Channels

S. VASUDEVAN[1] AND M. K. VARANASI[1], MEMBER, IEEE

Abstract

Synchronous Code Division Multiple Access (CDMA) data transmission over a non-selective Rician fading channel is considered, where the faded signal components arrive at the receiver in synchronism with the specular signal component. Under the assumption that the fading parameters are uncorrelated, this fading CDMA channel is shown to be equivalent to an additive Gaussian CDMA channel over which a modified signal set is employed. The fading channel characteristics, together with the signature signals employed over the fading channel determine the effective signature signals that characterize the equivalent non-fading CDMA channel. A fading counterpart of the multiuser asymptotic efficiency performance measure is defined and the results for non-fading CDMA channels are extended to this channel model. The conventional detector, which has low complexity, is shown to degrade rapidly in near-far environments. The optimal detector exhibits a considerable improvement in performance but at the price of a computational complexity that increases exponentially with the number of users. The decorrelating detector for the fading case is derived as a compromise. The asymptotic efficiencies of the fading detectors are seen to be bounded from above and below by the asymptotic efficiencies of their AWGN counterparts when the latter are used over the AWGN and faded channels, respectively. Numerical results for the two-user case indicate that, in a fading environment, the decorrelating and optimum fading detectors offer significant gains in performance when used instead of their AWGN counterparts.

1 Introduction

The problems of optimal as well as sub-optimal detection for CDMA transmission over an AWGN channel have been the focus of study in the recent past[1-9]. Given the fact that CDMA transmissions are frequently made over channels that exhibit fading and/or dispersion, it would seem appropriate to design receiver structures which take into account this behavior of the channels. In this paper, we focus attention on obtaining detection schemes for nondispersive channels that exhibit fading alone, in which the time or frequency dispersion is marginal, and yet the received signal is a significantly distorted version of the transmitted signal because each user's signal is amplitude modulated and phase shifted by a random and more likely different amount. Such channels have also been called (time) flat - (frequency) flat fading channels [10] since the fading, though random, does not vary over a signalling interval while affecting all the constituent frequencies in the transmitted signal in the same manner.

The rest of the paper is organized as follows. Section 2 lists out the notation used for the development in Section 3. In Section 3 the signal model is presented along with the optimum decision rule, under the assumption that the fading parameters affecting the various users' transmissions are uncorrelated. Further, the non-selective Rician fading CDMA channel with uncorrelated fading is shown to be equivalent to an AWGN CDMA channel over which a modified signal set is employed. The conventional, decorrelating and optimal detection strategies are analyzed in Section 4 along with a performance characterization using an asymptotic efficiency measure defined for the fading channel. The performance of the decorrelating and optimal detectors designed for the AWGN channel of [1], over the fading channel is analytically characterized and compared to that of their faded counterparts. A two-user example is considered in Section 5 and analytical characterizations as well as word descriptions of some graphical characterizations of the performance of the various detectors are presented. Section 6 outlines the principal conclusions and possible extensions.

2 Notation

$r(t)$: the received signal.

---

[1] The authors are with the department of Electrical and Computer Engineering at the University of Colorado, Boulder, CO 80309

$a_k(t)$: the k^{th} user's normalized signature signal, time limited to [0,T].
$f_k(t)$: the faded component of the k^{th} user's signal, time limited to [0,T].
$s_k(t)$: the specular component of the k^{th} user's signal, time limited to [0,T].
$n(t) = n_r(t) + jn_i(t)$: the complex, zero mean, additive,white Gaussian noise process.
σ^2 : the variance of the noise $n(t)$; $E[n_r(t)^2] = E[n_i(t)^2] = \frac{\sigma^2}{2}, E[n_r(t)n_i(t)] = 0$.
E_k : the energy in the specular component of the signature signal of the k^{th} user.
θ_k : the phase of the specular component of the signature signal of the k^{th} user.
γ_k : the Rayleigh distributed attenuation of the k^{th} user's signal; $E[\sqrt{E_k}\gamma_k]^2 = \sigma_k^2$.
ϕ_k : the random phase shift of the k^{th} user's signal, uniformly distributed over $[0, 2\pi]$.
$\mathbf{b} : [K \times 1]$, $b_k \in \{-1, +1\}$: the k^{th} user's data symbol.
$\psi : [K \times 1]$, $\psi_k = \gamma_k e^{j\phi_k}$.
$A : [K \times K]$, $A_{kl} = \int a_k(t)a_l^*(t)dt$.
$\Theta : [K \times K]$, $\Theta_{kl} = e^{j\theta_k}\delta_{k-l}$.
$\mathbf{q} : [K \times 1]$, $q_k = \int a_k(t)r^*(t)dt$.
$E : [K \times K]$, $E_{kl} = E_k\delta_{k-l}$.
$\Psi : [K \times K]$, $\Psi_{kl} = \gamma_k e^{j\phi_k}\delta_{k-l}$.
$\Sigma : [K \times K]$, $\Sigma_{kl} = \frac{\sigma_k}{\sigma}\delta_{k-l} = \sqrt{r}_k\delta_{k-l}$.
$\mathbf{x} : [\mathbf{m}, C]$, implies that $\mathbf{x}$ is a Gaussian random vector with mean $\mathbf{m}$ and covariance matrix C .
$*$ denotes complex conjugation.

3 Theory

The signal model we assume here is described by the equation

$$r(t) = \sum_{k=1}^{K} b_k s_k(t) + \sum_{k=1}^{K} b_k f_k(t) + n(t), \tag{1}$$

where the specular component $s_k(t)$, and the fading component $f_k(t)$, are given by the expressions $s_k(t) = \sqrt{E_k}a_k(t)e^{j\theta_k}$ and $f_k(t) = \sqrt{E_k}\gamma_k a_k(t)e^{j\phi_k}$, and the signal is observed over the interval $[0, T]$. In general, $a_k(t)$ is an arbitrary complex-valued signature waveform, thereby including spread-spectrum signalling schemes such as direct-sequence, frequency hopping, and a hybrid combination of both [7]. Typically, the specular (direct) component is associated with a clear path between the transmitter and receiver, while alternate paths along which the signal may be considerably weakened in strength give rise to the faded component [11]. In the present model we interpret E_k as the energy in the direct component of the signature signal of the k^{th} user, and define θ_k as the known phase of that component of the k^{th} user's signal at the receiver. γ_k and ϕ_k denote the Rayleigh distributed attenuation and uniformly distributed phase shift that the same user's signal undergoes along the fading path(s). We make the assumption that the fading parameters affecting the various users are independent of each other, justifying it by the fact that the paths travelled by the signals of these users are most likely different. Further, the signal set is assumed linearly independent, a constraint which is much weaker than orthogonality and feasible in physical implementations. A very similar model has been analyzed in [12] (and the references therein), assuming a conventional detection strategy. In more recent work on multiuser detection over AWGN channels [1-4], it has been demonstrated however, that conventional detectors are multiple-access limited, while the optimal and decorrelating detectors are not. It is in anticipation of similar improvements that we derive and analyze in this paper, in addition to the conventional detector, optimal and sub-optimal detection strategies for a fading environment. The following proposition states the optimum decision rule under the Maximum Likelihood (M L) criterion.

Proposition 1 *The optimum decision rule for detection of CDMA transmission over a non-selective Rician fading channel with uncorrelated fading is given by*

$$\widehat{\mathbf{b}} = \arg \max_{b \in \{-1,1\}^K} 2\mathbf{b}^T(\mathbf{y}_f + \mathbf{y}_f^*) - \mathbf{b}^T(H_f + H_f^*)\mathbf{b} \quad , \tag{2}$$

where

$$H_f = E^{\frac{1}{2}}\Theta(A^{-1} + \Sigma^2)^{-1}\Theta^* E^{\frac{1}{2}}, \tag{3}$$

and

$$\mathbf{y}_f = E^{\frac{1}{2}}\Theta(I - A(\Sigma^{-2} + A)^{-1})\mathbf{q}. \tag{4}$$

The problem of obtaining the maximizing $\mathbf{b}$ for the expression in (2) is NP hard, requiring an exhaustive search through all 2^K possible choices for the data vector. This follows from the fact that the functional form of the likelihood is the same as in [1].

Given the form of the solution to the detection problem over the Rician faded channel, one seeks a connection to the previously obtained solution [1] for optimal detection of CDMA transmission over an AWGN channel. At this stage we recall the detection problem formulation for the AWGN or non-faded channel:

$$\widehat{\mathbf{b}} = \arg \max_{\mathbf{b} \in \{-1.1\}^K} 2\mathbf{b}^T(\mathbf{y}_{nf} + \mathbf{y}^*_{nf}) - \mathbf{b}^T(H_{nf} + H^*_{nf})\mathbf{b}, \tag{5}$$

$$\begin{aligned} \mathbf{y}_{nf} + \mathbf{y}^*_{nf} &= (H_{nf} + H^*_{nf})\mathbf{b} + \mathbf{n}_{nf} + \mathbf{n}^*_{nf}, \\ \mathbf{n}_{nf} + \mathbf{n}^*_{nf} &: [0, \sigma^2(H_{nf} + H^*_{nf})], \end{aligned}$$

where $H_{nf} = E^{\frac{1}{2}}\Theta A \Theta^* E^{\frac{1}{2}}$ and $\mathbf{y}_{nf} = E^{\frac{1}{2}}\Theta\mathbf{q}$, where $\mathbf{q}$ is now the vector of normalized matched filter outputs obtained when fading is absent. The vector $\mathbf{y}_{nf} + \mathbf{y}^*_{nf}$ is the vector of sufficient statistics for the non-faded channel, while A and E are the normalized signal cross-correlation and energy matrices, respectively. The above representation is the low pass analogue of the passband description of the detection problem in [1]. The correspondences between the descriptions of (2) and (5), in order to apply the results derived in [1] to the case at hand, are made by the following proposition which makes the relationship between the faded and non-faded detection problems precise.

Proposition 2 *The non-selective Rician fading CDMA channel with uncorrelated fading is equivalent to a CDMA AWGN channel with signal parameters given by*

$$\begin{aligned} A_f &= E_d^{-\frac{1}{2}}(A^{-1} + \Sigma^2)^{-1}E_d^{-\frac{1}{2}}, \\ E_f &= E_d E, \end{aligned} \tag{6}$$

where A_f and E_f are the equivalent signal cross-correlation and energy matrices, and the elements of E_d are chosen such that $(A_f)_{kk} = 1$ for $1 \leq k \leq K$.

The previous result follows once we obtain the equations relating the vector $\mathbf{y}_f + \mathbf{y}^*_f$ to the data vector, making use of the fact that the fading parameters are uncorrelated with each other, and with the noise components produced at the receiver end by the additive white Gaussian noise processes, i.e., $E[\psi_k \psi^*_l] = \frac{\sigma_k^2}{E_k}\delta_{k-l}$ and $E[\psi_k n^*_l] = 0$ for $1 \leq k, l \leq K$ where $n_l = \int n(t)a^*_l(t)dt$. We have

$$\begin{aligned} \mathbf{q} &= A\Theta^* E^{\frac{1}{2}}\mathbf{b} + A\Psi^* E^{\frac{1}{2}}\mathbf{b} + \mathbf{n} && (7) \\ &= A\Theta^* E^{\frac{1}{2}}\mathbf{b} + \mathbf{n}_S && (8) \end{aligned}$$

where $\mathbf{n} : [0, \sigma^2 A]$, $A\Psi^* E^{\frac{1}{2}}\mathbf{b} : [0, \sigma^2 A\Sigma^2 A]$, and hence $\mathbf{n}_S$ has a normal distribution $[0, \sigma^2 A(\Sigma^2 + A^{-1})A]$ which is independent of $\mathbf{b}$.

Using (8) and (4) it may be verified that

$$\begin{aligned} \mathbf{y}_f + \mathbf{y}^*_f &= (H_f + H^*_f)\mathbf{b} + \mathbf{n}_f + \mathbf{n}^*_f, \\ \mathbf{n}_f + \mathbf{n}^*_f &: [0, \sigma^2(H_f + H^*_f)]. \end{aligned} \tag{9}$$

Inspection of (2), (3), (9), and (5) reveals that the vector of sufficient statistics: $\mathbf{y}_f + \mathbf{y}^*_f$, for the data transmitted over the fading channel may now be seen as a vector of sufficient statistics for the data assuming it was transmitted across an AWGN channel using a signal set whose normalized cross-correlations and energies are given by the normalized positive definite matrix A_f and the diagonal matrix E_f respectively, which are defined by (6). Fig. 1 illustrates the equivalence stated above. The cross-correlation matrix of the signals $\{a'_1(t), \cdots, a'_K(t)\}$ used by transmitters $\{T'_1, \cdots, T'_K\}$ in the equivalent CDMA-AWGN channel is A_f, and the transmissions are made with energies $\{(E_f)_{kk}\} |_{k=1}^K$. The complex conjugate of the channel output in each case is correlated with the signals shown and the statistics are obtained after the appropriate operations. The following statements about the behavior of the effective energies and correlations are easily proved.

Corollary 1 *For non-zero variance ratios, the effective signal energies in the equivalent AWGN-CDMA channel are all less than the corresponding actual specular signal energies.*

Corollary 2 *The absolute value of the correlation between any two signals belonging to the set specifying the equivalent non-fading system, cannot increase with an increase in the fading to additive noise variance ratios associated with either user.*

Intuitively, the correlation between the signals in the equivalent signal set might be expected to increase with increasing fading strength (and σ^2 fixed). However, the converse occurs. In fact, in the limit as the fading levels become infinite we have an equivalent signal set that is orthogonal. The catch of course is that these signals also have zero energy.

4 Analysis of Detection Strategies

In this section, we outline the conventional, decorrelating and optimal detection strategies for the fading channel. These strategies are also compared with those designed for the CDMA-AWGN channel, when the latter are employed over the fading channel. The nature of the fading channel determines in part the choice and definition of one or more performance measures. The asymptotic efficiency of the k^{th} user of a particular detection strategy has been defined for a CDMA transmission over an AWGN channel as [3]

$$\eta_k = \sup\{0 \leq r \leq 1; \lim_{\sigma \to 0} \mathcal{P}_k(\sigma)/Q\left(\frac{\sqrt{r w_k}}{\sigma}\right) < +\infty\}, \tag{10}$$

where $\mathcal{P}_k(\sigma)$ is the k^{th} user's error probability in the multiuser environment , σ is the standard deviation of the additive noise and w_k is the energy in the k^{th} user's signature signal. The term in the denominator is the error probability for coherent (optimum) detection of antipodal data transmitted over an AWGN channel by a single user with signal energy equal to $r w_k$ when the additive noise variance equals σ^2. In choosing the same definition of asymptotic efficiency for CDMA transmission across a fading channel with only the additive noise variance going to zero, we find that by this criterion, all detection schemes over a fading channel would have zero asymptotic efficiency. To quantify the multiple-access and fading limitations of the various detection schemes, we let the fading variances decay at the same rate as the additive noise variance. We thus state fading counterparts of the results obtained for data transmission over the AWGN channel [1] using the following definitions of asymptotic efficiency for the k^{th} user :

$$\eta_k^{f1} = \sup\ \{0 \leq r \leq 1; \lim_{\sigma \to 0, \{\frac{\sigma_l}{\sigma} = \sqrt{r_l}\}} \mathcal{P}_k(\sigma, \{\sigma_l\})/Q\left(\frac{\sqrt{2rE_k}}{\sigma}\right) < +\infty\}, \tag{11}$$

$$\eta_k^{f2} = \sup\ \{0 \leq r \leq 1; \lim_{\sigma \to 0, \{\frac{\sigma_l}{\sigma} = \sqrt{r_l}\}} \mathcal{P}_k(\sigma, \{\sigma_l\})/Q\left(\frac{\sqrt{2rE_k}}{\sqrt{\sigma^2 + \sigma_1^2}}\right) < +\infty\}, \tag{12}$$

where the (finite) fading to additive noise variance ratios $\{r_l\}\ |_{l=1}^{K}$, are held constant, and the specular component energies are assumed functionally independent of the fading variances. The first definition compares the multiuser error probability to that of the single user in AWGN, while the second makes a comparison with the single user error probability in a faded environment. The above measures capture the performance limitations due the presence of the other users at fixed variance ratio levels in the high SNR and high signal to fading noise ratio (SFNR) regions, which are the regions of interest in our analysis.

The near-far resistance ($\overline{\eta}_k$) of a user employing a particular detection strategy, has been defined [1] as the infimum of the asymptotic efficiency of that user, over the energies of all the other interfering users:

$$\overline{\eta}_k = \inf_{w_l \geq 0, l \neq k} \eta_k. \tag{13}$$

It is a measure of the robustness of a detection scheme to variations in the energies of the users, and detectors with non-zero near-far resistance are called near-far resistant. The definition (13) for the near-far resistance holds for the fading channel as well, with the minimization (over the specular component energies) now being carried out of one of the two asymptotic efficiency measures defined by (11) and (12).

The use of the performance measures (11) and (13), for a characterization of the various detectors designed for the fading channel as well as their AWGN counterparts, yields a comprehensive set of results which makes it possible to compare the schemes analytically.

4.1 Conventional Detection

The optimum decision rule in a single user fading environment chooses the data symbol $b \in \{-1, 1\}$ as

$$\begin{aligned} \widehat{b} &= sgn\ (y_f + y_f^*) \\ &= sgn\ \sqrt{E_1}(\exp^{j\theta_1} q + \exp^{-j\theta_1} q^*), \end{aligned}$$

where $q = \int a_1(t) r^*(t) dt$, $a_1(t)$ is the user's normalized signature signal, $r(t)$ is the recieved signal, and θ_1 and E_1 are the phase and energy of the specular signal component respectively. The single user decision rule applied in the multiuser system constitutes the conventional detector. The conventional decision rule for each user is thus obtained by assuming the absence of the other users and turns out in the multiuser environment to be a sign test on the vector $\widetilde{\mathbf{y}}_{nf} + \widetilde{\mathbf{y}}_{nf}^*$ where

$$\begin{aligned} \widetilde{\mathbf{y}}_{nf} &= E^{\frac{1}{2}} \Theta \mathbf{q} \\ &= H_{nf} \mathbf{b} + \widetilde{\mathbf{n}}. \end{aligned} \tag{14}$$

In stating the second equality, we have made use of (8) and defined $\widetilde{\mathbf{n}} = E^{\frac{1}{2}} \Theta \mathbf{n}_S$. Clearly, $\widetilde{\mathbf{n}}$ is a normal random vector with distribution $[0, \sigma^2 \widetilde{H}]$ where

$$\widetilde{H} = E^{\frac{1}{2}} \Theta (A \Sigma^2 A + A) \Theta^* E^{\frac{1}{2}}.$$

Hence, the probability of error in the k^{th} bit, for this detector is given by

$$\begin{aligned} \mathcal{P}_k^c &= \mathcal{P}(\widehat{b}_k = 1 / b_k = -1) \\ &= 2^{1-K} \sum_{\mathbf{b}, b_k = -1} Q\left(- \frac{\sum_l ((H_{nf} + H_{nf}^*)_{kl} b_l)}{\sigma \sqrt{(\widetilde{H} + \widetilde{H}^*)_{kk}}} \right). \end{aligned} \tag{15}$$

The asymptotic efficiency of the k^{th} user of the conventional detector may now be obtained by using the bit error probability expression (15) and the definition (11) . We have

$$\eta_k^{f_1} = \max{}^2 \left\{ \frac{1}{\sqrt{2E_k}} \frac{(H_{nf} + H_{nf}^*)_{kk} - \sum_{l \neq k} \mid (H_{nf} + H_{nf}^*)_{kl} \mid}{\sqrt{(\widetilde{H} + \widetilde{H}^*)_{kk}}}, 0 \right\}. \tag{16}$$

The result follows on realizing the dominance of a single term corresponding to the worst-case interfering bit combination, in the summation of (15), in the high SNR-SFNR region.

Given the functional dependence of $\eta_k^{f_1}$ on the interfering energy ratios $\{E_l / E_k\} \mid_{l \neq k}$, we may use the idea of the corresponding proof in [1] to show that the near-far resistance of this detector is zero, except when the signal of the user in question is orthogonal to the subspace spanned by the signals of the other K−1 users.

4.2 Linear Detection

4.2.1 Fading Decorrelator

The decorrelating detector which is an energy-independent linear detection strategy, was analyzed in [1] for the AWGN channel. The detector was derived by first treating the energies in the likelihood function as unknowns along with the data bits. Under the assumption that the fading parameter variances are independent of the direct component energies, we may obtain an easily implementable linear detector for use over the fading channel. Such an assumption would be justified in situations, where the channels across which the specular and faded components travel are very different, and the channels are primarily responsible for variations in the energies received at the detector. The form of the decorrelating solution for the fading channel may be obtained by maximizing the likelihood function of (2) with respect to $E^{\frac{1}{2}} \mathbf{b}$. The solution is given by

$$\begin{aligned} \widehat{E^{\frac{1}{2}} \mathbf{b}} &= (R_f + R_f^*)^{-1} (\mathbf{y}_{fd} + \mathbf{y}_{fd}^*), \\ \widehat{\mathbf{b}} &= sgn\ (R_f + R_f^*)^{-1} (\mathbf{y}_{fd} + \mathbf{y}_{fd}^*), \end{aligned} \tag{17}$$

where $R_f = \Theta(A^{-1}+\Sigma^2)^{-1}\Theta^*$, and $\mathbf{y}_{fd} = \Theta(I - A(\Sigma^{-2}+A)^{-1})\mathbf{q}$. The probability of bit error for the k^{th} user decorrelator may be obtained by using the fact that the estimator defined by (17) is a normally distributed vector. Hence

$$\mathcal{P}_k^{fdf} = Q\left(\frac{1}{\sigma\sqrt{((H_f + H_f^*)^{-1})_{kk}}}\right) \tag{18}$$

with H_f defined as before. In light of the previous results we may state the following properties concerning the performance of the decorrelator.

1. *The decorrelator has an energy-independent k^{th} user asymptotic efficiency equal to $1/2((R_f + R_f^*)^{-1})_{kk}$ which is therefore equal to its near-far resistance.*

The performance of the decorrelator for a particular user is thus invariant to changes in the energy levels of the other users. The near-far problem which refers to the deterioration in detector performance when the energies of the interfering users increase, is thus alleviated. One observes however, that the performance of this detector is still a function of the specular signal phases.

2. *The faded decorrelator achieves the highest achievable near-far resistance.*

The near-far resistance (or worst case asymptotic efficiency) of the optimal fading detector is seen to be equal to the asymptotic efficiency (or near-far resistance) of the fading decorrelator, from Proposition 4. The fading decorrelator is thus optimally near-far resistant.

4.2.2 AWGN Decorrelator

It is worthwhile to consider the performance of the decorrelating detector designed for the AWGN channel (where only the specular components with energies $\{E_l\}\,|_{l=1}^{K}$ are present), in a fading environment, since one would like to quantify the loss incurred by ignoring the presence of the faded signal components. We recall that the non-fading decorrelator produces the following estimate of the data vector $\mathbf{b}$:

$$\begin{aligned} \widehat{\mathbf{b}} &= sgn\ \ (\Theta A\Theta^* + \Theta^* A^* \Theta)^{-1}(\Theta\mathbf{q} + \Theta^*\mathbf{q}^*) \\ &= sgn\ \ (R_{nf} + R_{nf}^*)^{-1}(\Theta\mathbf{q} + \Theta^*\mathbf{q}^*)\ , \end{aligned} \tag{19}$$

where $\mathbf{q}$ is the vector of normalized matched filter outputs. When this detector is used over the faded channel, the probability of an error in the k^{th} bit is given by

$$\mathcal{P}_k^{df} = Q\left(\frac{E_k}{\sigma\sqrt{((R_{nf}+R_{nf}^*)^{-1}(\widetilde{R}+\widetilde{R}^*)(R_{nf}+R_{nf}^*)^{-1})_{kk}}}\right), \tag{20}$$

where $\widetilde{R} = E^{-\frac{1}{2}}\widetilde{H}E^{-\frac{1}{2}}$, from which results an energy-independent expression for the asymptotic efficiency using (11). The asymptotic efficiency of the same user of this detector is found to be

$$\begin{aligned} \eta_k^{f_1} &= \frac{1}{2}\frac{1}{((R_{nf}+R_{nf}^*)^{-1}(\widetilde{R}+\widetilde{R}^*)(R_{nf}+R_{nf}^*)^{-1})_{kk}} \\ &> 0\ . \end{aligned} \tag{21}$$

The second inequality is a consequence of the positive definiteness of the matrices involved, and the assumption of finite variance ratios. Since $\eta_1^{f_1}$ is energy independent in this case, it is also the near-far resistance of the user. Thus it follows that both the AWGN and the fading decorrelators are near-far resistant.

4.2.3 Performance Comparison

In this subsection we compare analytically the probabilities of error of the faded and AWGN decorrelators when the former is used in the environment it is designed for, and the latter is implemented over either channel. We recall that the AWGN decorrelator, when used over the AWGN channel, has a probability of error in the k^{th} bit given by

$$\mathcal{P}_k^{d} = Q\left(\frac{1}{\sigma\sqrt{((H_{nf} + H_{nf}^*)^{-1})_{kk}}}\right). \tag{22}$$

The following proposition then, orders the decorrelating detectors according to their performance.

Proposition 3 *The probability of bit error of the fading decorrelator in a fading environment is upper bounded by that of the AWGN decorrelator in the same environment, and lower bounded by the probability of bit error of the AWGN decorrelator when the latter is used over the AWGN channel:*

$$\mathcal{P}_k^{d} \quad \le \quad \mathcal{P}_k^{fdf} \quad \le \quad \mathcal{P}_k^{df} \qquad\qquad k \in \{1, 2, \cdots, K\}.$$

4.3 Optimal Detection

4.3.1 Optimum Fading Detector

The optimum detector obtains the data sequence that maximizes the likelihood function derived in Section 3. Due to its complexity, it is an unlikely candidate for implementation, particularly when the number of users is large. However, the performance of this detector serves as a basis for determining the usefulness of various sub-optimal detectors. The characterization of the performance of this detector in the high SNR - SFNR region requires upper and lower bounds on the bit error probability which are asymptotically tight in the sense that both the upper and lower bounds exhibit the same exponential decay. As a consequence of Proposition 2, one may view the error probability analysis of the faded optimal detector as an application of the bounds of [2] to error analysis of an optimal detector of CDMA transmissions in an AWGN channel. Asymptotic tightness of these bounds may be ensured by letting the fading variances decay at the same rate as the additive noise variance leading to the following results pertaining to the asymptotic efficiency and near-far resistance, in the fading situation.

Proposition 4 *The asymptotic efficiency of the k^{th} user employing the optimum detector is given by*

$$\eta_k^{f_1} \quad = \quad \frac{1}{2E_k} \min_{\underline{\epsilon}\in\{-1,0,1\}^K, \epsilon_k=1} \underline{\epsilon}^T (H_f + H_f^*)\underline{\epsilon} \ . \tag{23}$$

The near-far resistance of that user is given by

$$\overline{\eta}_k^{f_1} \quad = \quad \frac{1}{2((R_f + R_f^*)^{-1})_{kk}} \ . \tag{24}$$

The above value of near-far resistance (worst case asymptotic efficiency) is achieved by the fading decorrelator.

4.3.2 Optimal AWGN Detector

The optimal non-fading detector maximizes the likelihood function (5). It ignores the presence of fading assuming the transmission of only the direct signal components. It then uses the transmissions that occur over the faded channel to obtain the sufficient statistics for the AWGN channel. These are then used in the likelihood function obtained for the non-faded channel, and the optimizing data vector found. It turns out that asymptotically tight bounds on the probability of bit error of this detector may be stated under the assumption that the fading variances decay at the same rate as the additive noise variance. This results in an expression for the asymptotic efficiency of this detector.

Proposition 5 *The asymptotic efficiency of the k^{th} user of the optimal AWGN detector in the faded environment is given by*

$$\eta_k^{f_1} \quad = \quad \frac{1}{2E_k} \min_{\underline{\epsilon}\in\{-1,0,1\}^K, \epsilon_k=1} \frac{(\underline{\epsilon}^T (H_{nf} + H_{nf}^*)\underline{\epsilon})^2}{\underline{\epsilon}^T (\widetilde{H} + \widetilde{H}^*)\underline{\epsilon}} \ . \tag{25}$$

We observe that, as a consequence of the positive definiteness of $R_{nf} + R_{nf}^*$, and the assumption of finite fading to additive noise variance ratios, the detector is near-far resistant. Additionally, the following corrollary holds.

Corollary 3 *The near-far resistance of the k^{th} user of the optimal AWGN detector is upperbounded by that achieved by the same user of the AWGN decorrelator.*

We offer the following heuristics to account for the previous corollary. The presence of fading may be seen as altering both the energies and correlations of the transmitted signal set. While the AWGN decorrelator and optimal AWGN detector both suffer losses in performance in the faded environment due to the changed correlations and energies, the effect on the latter is more severe since it uses incorrect values of both these quantities in making decisions. The AWGN decorrelator is relatively more robust since it does not use information about the direct component energies of the users' tranmissions.

4.3.3 Performance Comparison

In this subsection we compare analytically the asymptotic efficiencies of the optimal faded and AWGN detectors when the former is used in the environment it is designed for, and the latter is implemented over either channel. We recall that the optimum AWGN detector, when used over the AWGN channel, has a k^{th} user asymptotic efficiency given by

$$\eta_k^{f_1} \quad = \quad \frac{1}{2E_k} \min_{\underline{\epsilon}\in\{-1,0,1\}^K, \epsilon_k=1} \underline{\epsilon}^T (H_{nf} + H_{nf}^*)\underline{\epsilon}. \tag{26}$$

The following proposition orders the optimal detectors according to their performance.

Proposition 6 *The k^{th} user asymptotic efficiency of the optimum fading detector in a fading environment is lower bounded by that of the optimum AWGN detector in the same environment, and upper bounded by the k^{th} user asymptotic efficiency of the optimum AWGN detector when the latter is used over the AWGN channel.*

5 A Two-User Example

In this section we consider the changes in the two-user system performance due to channel parameter (r_1, r_2) variations as well as variations in specular signal strengths. In this example, r_1 and r_2 are the variance ratios associated with the two users and $\theta = \theta_2 - \theta_1$ is the phase difference between their specular signals. Further E_1 and E_2 are the respective specular component energies, and the cross-correlation $\rho \neq 0$ between the users' normalized signals is assumed real.

Particularized closed form expressions for, and relationships between, the asymptotic efficiencies for the various fading detectors are presented below. The asymptotic efficiency for the first user of the conventional detector is given by

$$\eta_1^{f_1} \quad = \quad \max{}^2 \left\{ \frac{1- \mid \rho\cos\theta \mid \sqrt{\frac{E_2}{E_1}}}{\sqrt{r_1 + \rho^2 r_2 + 1}}, 0 \right\}. \tag{27}$$

When the relative signal strength of the interfering user is sufficiently high, and in particular, if $\sqrt{\frac{E_2}{E_1}} \geq \frac{1}{|\rho\cos\theta|}$, then the asymptotic efficiency is zero. This illustrates the near-far problem that the conventional detector suffers from; the error probability of the detector approaching one-half for sufficiently large interfering energy ratios. The first user asymptotic efficiency (and near-far resistance) using the decorrelator is given by

$$\eta_1^{f_1} \quad = \quad \frac{1-\rho^2}{1+r_1(1-\rho^2)} \frac{((1+r_1(1-\rho^2))(1+r_2(1-\rho^2)) - \rho^2\cos^2\theta)}{((1+r_1(1-\rho^2))(1+r_2(1-\rho^2)) - \rho^2)}. \tag{28}$$

As the fading levels decrease, the asymptotic efficiency increases so that we have, in the limit

$$\lim_{r_1, r_2 \to 0} \eta_1^{f_1} \quad = \quad 1 - \rho^2\cos^2\theta, \tag{29}$$

which is the asymptotic efficiency of that user of the AWGN decorrelator in a non-fading environment. Finally, the first user of the optimal detector has an asymptotic efficiency defined by

$$\begin{aligned} \eta_1^{f_1} \quad = \quad \min \Bigg[& \frac{1}{2}(R_f + R_f^*)_{11}, \\ & \frac{1}{2}((R_f + R_f^*)_{11} - \mid (R_f + R_f^*)_{21} + (R_f + R_f^*)_{12} \mid \sqrt{\frac{E_2}{E_1}} + (R_f + R_f^*)_{22}\frac{E_2}{E_1}) \Bigg], \end{aligned} \tag{30}$$

where $R_f + R_f^*$ is given by

$$R_f + R_f^* \quad = \quad \frac{2(1-\rho^2)}{(1+r_1(1-\rho^2))(1+r_2(1-\rho^2)) - \rho^2} \begin{bmatrix} (1+r_2(1-\rho^2)) & \rho\cos\theta \\ \rho\cos\theta & (1+r_1(1-\rho^2)) \end{bmatrix}.$$

This asymptotic efficiency achieves its minimum value at $\sqrt{\frac{E_2}{E_1}} = \frac{|\rho\cos\theta|}{1+r_1(1-\rho^2)}$. The value of $\eta_1^{f_1}$ obtained at this point, using (30), equals the asymptotic efficiency obtained for the fading decorrelator's first user given by (28).

The following is a description of the general behavior of the various detectors as a function of the variance, and energy ratios, as observed from plots of the asymptotic efficiency measures. The asymptotic efficiency measure obtained by comparison with a non-faded single-user system (11) is our basis for quantifying degradation in detector performance in the high SNR-SFNR region. The plots (Fig. 2 is an example) for the two-user case illustrate the deterioration in the performance of both the optimal and conventional detectors as the fading levels increase. In either case, the effect of an increase in a variance ratio is more pronounced on the user associated with it. One notes also that it is only the conventional detector that is multiple-access limited. Additionally one observes that the conventional detector achieves the same performance as the optimal detector at the zero interference-energy point, when the variance ratio associated with the interfering user is zero. Finally, one also notes from Fig. 2 the fact that the fading decorrelator is optimally near-far resistant.

The plot obtained by using (12) as the definition of asymptotic efficiency is seen to show higher values of η_1^{f2} for both conventional and optimum detectors at higher fading levels. Viewed in light of the results observed by using η_1^{f1}, the conclusion is that fading has a greater relative effect on a single-user system.

Finally the performance loss when the non-fading optimal detector is used instead of its fading counterpart, in the fading environment has been observed. In the high interference energy region, the first user of the optimal AWGN detector achieves the same performance with the optimum detector when the variance ratio associated with the other user is zero. In general though, the optimum detector outperforms its AWGN counterpart, the difference increasing with this variance ratio. A similar situation prevails in the case of the AWGN and fading decorrelators.

6 Conclusions

In this paper we have obtained the optimum detector for a synchronous CDMA system with the signals being transmitted across a non-selective Rician fading channel in which the fading is uncorrelated, while simultaneously being corrupted additively by white Gaussian noise. Under the same assumptions, it was shown that the optimum detection problem could be described in a manner analogous to the non-fading synchronous CDMA detection problem in additive, white Gaussian noise (in effect establishing the existence of an equivalent non-fading problem). The main consequences of this equivalence were that the users could now be seen as transmitting data using a new set of signature signals with reduced energy. Also, a characterization of the limitations in the performance of various detectors due to multiple-access interference and fading, similar to the development in the AWGN channel became possible. Hence, asymptotic efficiency measures were defined to characterize the performance of the various detectors as well as their AWGN counterparts, in a fading environment. The performance levels, as characterized by the asymptotic efficiencies and probabilities of error attained by the optimal fading detector and decorrelator respectively, were found to lie between those achieved by their AWGN counterparts over the fading and AWGN channels.

The decorrelating and optimal detectors designed for the fading channel were found to be near-far resistant, while the conventional detector was not. The near-far problem is thus not a problem endemic to CDMA communication over fading channels. In this context, one notes that the AWGN decorrelator as well as the optimal AWGN detector were found to be near-far resistant too. These detectors however, achieved in general, lower asymptotic efficiencies than their fading counterparts. The numerical results of Section 5 for the two-user case point to the gains that accrue from replacing the AWGN optimal and decorrelating detectors by their fading counterparts, in a fading environment. Expectedly, larger gains are made when the variance ratios are large. The availability of closed form expressions for the asymptotic efficiencies of each of these detectors allows for a quantification of this improvement for the K-user case too.

The results of this paper also illustrate the fact that the CDMA-AWGN signal model extended to a simple fading situation leads to significant changes in detector implementations. A consequence of the significant gains observed on substituting the AWGN detectors by their fading counterparts, is that the study of detection strategies for more complex fading and/or dispersion models in an asynchronous setting is motivated. Thus future directions would include generalizations such as the case where the specular and multiple faded components arrive at the reciever asynchronously as do the specular components of the signals of the different users of the system.

7 References

1. R. Lupas and S. Verdu. *Linear Multiuser Detectors for Synchronous Code-Division Multiple-Access Channels*, IEEE Trans. on Inf. Theory, Vol. 35, No. 1, pp 123-136, January 1989.
2. S. Verdu. *Minimum Probability of Error for Asynchronous Gaussian Multiple-Access Channels*, IEEE Trans. on Inf. Theory, Vol. 32, No. 1, pp 85-96, January 1986.
3. S. Verdu. *Optimum Multiuser Asymptotic Efficiency*, IEEE Trans. on Comm., Vol. 34, No. 9, pp 890-897, September 1986.
4. R. Lupas and S. Verdu. *Near-Far Resistance of Multiuser Detectors in Asynchronous Channels*, IEEE Trans. on Comm., Vol. 38, No. 4, pp 496-508, April 1990.
5. M. K. Varanasi and B. Aazhang. *Multistage Detection for Asynchronous Code-Division Multiple-Access Communications*, IEEE Trans. on Comm., Vol. 38, No. 4, pp 509-519, April 1990.
6. M. K. Varanasi and B. Aazhang. *Near-optimum Detection in Synchronous Code-Division Multiple Access Systems*, IEEE Trans. on Comm., Vol. 39, No. 5, pp 725-736, May 1991.
7. M. K. Varanasi and B. Aazhang. *Optimally Near-Far Resistant Multiuser Detection in Differentially Coherent Synchronous Channels*, IEEE Trans. on Info. Thry., Vol. 37, No. 4, pp 1006-1018, July 1991.
8. Z. Xie, C. R. Rushforth, and R.T. Short. *Multiuser Signal Detection Using Sequential Decoding*, IEEE Trans. on Comm., Vol. 38, No. 5, pp 578-583, May 1990.
9. Z. Xie, R. T. Short, and C. K. Rushforth. *A Family of Suboptimum Detectors for Coherent Multiuser Communications*, IEEE Journal on Selected Areas in Comm., Vol. 18, No. 4, pp 683-690, May 1990.
10. R. S. Kennedy. *Fading Dispersive Communication Channels* , Wiley-Interscience 1969.
11. D. E. Borth and M. B. Pursley. *Analysis of Direct-Sequence Spread-Spectrum Multiple-Access Communication over Rician fading Channels*, IEEE Trans. on Comm., Vol. 27, No. 10, pp 1566-1577, October 1979.
12. E. A. Geraniotis. *Direct-Sequence Spread-Spectrum Multiple-Access Communications over Non-Selective and Frequency-Selective Rician Fading Channels*, IEEE Trans. on Comm., Vol. 34, No. 8, pp 756-764, August 1986.

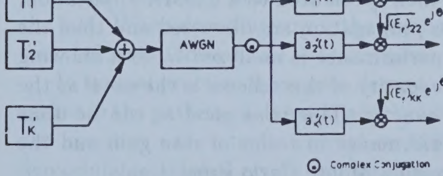


Figure 1. Equivalence of Faded and AWGN CDMA channels.

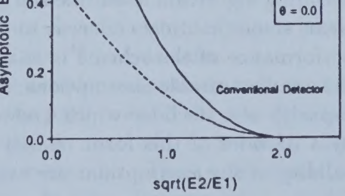


Figure 2. $\eta_1^{f_1}$ vs $\sqrt{\frac{E_2}{E_1}}$ for the conventional, faded optimal, and decorrelating detectors.

Exploiting Error Dependence in a Slow-Frequency-Hopped CDMA System

M.S. McKinley and W.E. Stark
Electrical Engineering and Computer Science Department
University of Michigan
Ann Arbor, MI 48109-2122

Abstract

An error-control coding scheme is described for use on channels with memory. The scheme uses a sequence of coupled, block-code decoders. The application of the scheme to a slow-frequency-hopped CDMA system using DQPSK modulation is described. Both analysis and simulation are used to calculate performance and investigate modeling and error propagation questions.

1. Introduction

In this paper an error-control coding scheme is described for use on channels with memory. The underlying idea behind this scheme is to exploit the correlation in the error process at the decoder input rather than ignore it via interleaving. For the same average error probability, the capacity of a channel with memory is higher than one without memory [1]. Thus ignoring the channel memory causes a loss in possible decoder performance.

In a code-division multiple-access (CDMA) system based on frequency hopping, the error correlation is introduced by the multiple access (MA) interference. The symbol error probability is higher when a symbol is hit by an interfering user than when noise alone corrupts the symbol. Since this hit process is very correlated from one symbol in a hop to the next, the error process also shows correlation.

The scheme described here passes information between successive decoders as was described in [2] and elsewhere. It can also be viewed as a generalization of the use of "test bits" as in [3, 4, 5].

In what follows the system in which the decoder is to be evaluated is described, the decoding algorithm is outlined in general and then specialized to a CDMA application. Next, some practical concerns to do with error propagation are discussed and then the performance of the scheme is analyzed. The performance is analyzed by first showing that under suitable assumptions, the limiting capacity of this scheme is the same as the capacity of a decoder which knows the state exactly. This is as good as can be done by a decoder of this form. Lastly, system performance in terms of rate gain and the validity of the assumptions are evaluated through a Monte Carlo simulation.

2. The System

The system considered here can be divided into four functional blocks: transmitter, channel, demodulator and decoder. The demodulator and decoder together are called the receiver.

The transmitter emits block-coded, slow-frequency-hopped DQPSK with mutually statistically independent information and hop frequency sequences. The transmitted symbols are formated into a block of codewords called a frame (viewed as a rectangular array). The rows of each frame are codewords from an error-correcting code. Each column of a frame is transmitted on a single hop in the normal manner of an interleaved system. Thus only one symbol from each codeword falls in each hop.

The demodulator simply makes independent, hard-decisions.

There are two channels considered. The first, called the conditionally-independent-error (CIE) channel, models the symbol error process at the demodulator output (decoder input) as being conditionally independent given the number of interferers present during each symbol interval. That is, each demodulator symbol-decision will be wrong with probability p_k for $k \geq 0$ when there are k interferers present. These errors are independent from one symbol to the next given k.

The other channel considered, called the equal power interferer (EPI) channel, consists of additive interference from k interferers. Each interferer uses the same signal format, but has independent information, carrier phase and hop frequency sequences. All interferers are received with the same power as the desired signal.

The first channel model is intended as an analytically tractable approximation to the second more realistic model. The accuracy of this approximation will be verified via simulation.

The received signal (demodulator input) is processed in three steps. First, the demodulator makes independent, hard-decisions on each of the received symbols in the received frame array. Next, the frame decoder will row-by-row decode this array of hard-detected symbols. Each row-decoder uses information from previously decoded rows via an errors-and-erasures decoding algorithm. Lastly, the original hard-decision frame estimate from the demodulator and the frame decoder output will be compared and a confidence assigned to the decoded frame based on the probability of the apparent error pattern between these two. This frame confidence is not part of the decoding algorithm, but could be used to monitor potential error propagation problems and/or to signal the retransmission of the frame in an ARQ system.

2.1 The Decoding Algorithm

Frame decoding operates on the hard-decision symbol estimates from the demodulator. No side information about the interference is used.

Frame decoding proceeds on a row-by-row basis from the first row in the frame to the last. When decoding row i, the demodulator and row-decoder outputs for previously decoded rows (rows 1 to $i-1$) comprise the state for the i^{th} row-decoder.

In typical practice these past demodulator and row-decoder outputs are only used to estimate the pattern of demodulator symbol errors in the prior rows. If the row-decoders are assumed either to be correct or fail (never output an incorrect codeword), then demodulator symbol errors can be detected by the row-decoder. Thus a demodulator error is declared whenever the demodulator and row-decoder disagree on a symbol value. The state typically comprises only this pattern of symbol error, no-error and failure estimates in prior row decodings.

Upon receipt and demodulation of a frame, the first row is decoded with no information about the state of the channel. Next, the second row of the frame is decoded using the demodulator output from the second row as input. The demodulator and decoder outputs from the first row are used as state to aid in this decoding.

In general, to decode row i, the demodulator output from row i is used as input and the demodulator and decoder outputs from rows 1 to $i-1$ are used as state. This state aids in the decoding of row i. This pattern continues to the last row in the frame.

To decode row i, the state for row i is used to estimate the error probability for each of the row-decoder input symbols (the hard-detected demodulator outputs for the row), given the state. These probabilities are used as the "channel measurement information" in a Chase [6] decoding scheme to form the row decoding for row i.

In a frequency-hopping CDMA application, the columns of the frame (the hops) are mutually independent. Thus the error probability at the demodulator output for the k^{th} symbol in the i^{th} row depends only on the k^{th} column (hop). Thus, the decoder for row i uses only that part of the state from the k^{th} hop to estimate the demodulator error probability for the k^{th} symbol in the row. When this has been done for each symbol in the row, the hard-detected demodulator output for the i^{th} row and these error probability estimates are passed to a Chase decoder to decode the row. This process continues from the first row (which has no state information) to the last row in the frame.

After frame decoding, an estimate can be made of the probability of the apparent error pattern between the demodulator and decoder outputs for the frame. This probability can be used as a measure of the confidence of the frame decoding.

A potential problem with the proposed decoding algorithm is error propagation. That is, when prior row decoding errors are made the estimated state of the channel is wrong. This makes the current and future decodings more likely to be in error.

When Reed-Solomon codes are used for the row codes and the unerased distance (the code minimum distance excluding erased positions) of the row code is large enough, then the probability of row-decoder error is much smaller than the probability of row-decoder failure. Thus, the majority of times that too many errors and/or erasures occur in a row, a failure will be detected instead of an error. In this case the state for successive row decodings will be less complete since the failed row contributes less information. The decoding algorithm will handle row decoding failures, and simulation will later be used to investigate possible error propagation.

In the analysis that follows, it will be assumed that the state is never wrong (it is either correct or contains failures). In practice, error propagation problems could be handled by using powerful, low rate codes periodically within the frame, and/or the estimated frame reliability as an overall decoding check.

3. Limiting Channel Capacity

Under suitable assumptions the limiting capacity of our decoding scheme, which estimates the symbol error probability, is equal to the capacity of a decoder which is given this knowledge. This is as good as can be done with a decoder of this form. We will show that as the number of symbols in a hop is increased it is possible to estimate the symbol error probability to any accuracy. The CIE channel will be used, and it will be assumed that the past row decodings are correct.

We will operate here on the M^{th} (last) row in the frame and then let $M \to \infty$ to get the limiting row capacity. We will later show that this capacity dominates the capacity of the whole decoder as $M \to \infty$.

Let k be the number of interfering users; p_k = the symbol error probability at the demodulator output given there are k interfering users $k \geq 0$. We assume that $p_k \leq p_{k+1}$.

Let q be the number of hop frequencies;

$\mathbf{S}^M$ = the binary state vector of length $M-1$ whose elements represent either error or no-error for the symbols before the M^{th} (last) symbol in a hop. $\mathbf{S}_i^M = 1, 1 \leq i \leq M-1$ if there is an error at symbol number i and 0 else (note that a ternary state which also includes the possibility of a row failure will be used later);

E_M = the event that the hard-detected value of the M^{th} demodulated symbol in a hop is in error (this is before decoding);

W_M = a window of symbols of length W in the desired user's hop between symbol numbers $M - W + 1$ and M. We must have $1 < W < M$;

We are interested in showing that we can estimate the error probability in the M^{th} symbol interval of each hop using the decoder state $\mathbf{S}^M$. To do this we will use $\hat{P}(E_M)$ which is defined as

$$\hat{P}(E_M) = \frac{1}{W-1} \sum_{i=M-W+1}^{M-1} \mathbf{S}_i^M$$

We now consider two cases. In both cases there are k interfering users present during symbol number M. Let T_{W_M} be the event that one or more interfering users hop into or out of the reference user's frequency slot during the window W_M. Consider the interference case where $\overline{T}_{W_M}$ occurs (overbar means event complement). Since there are no interference transitions, the number of interfering users and thus the error probability is constant during W_M. By the weak law of large numbers,

$$\lim_{\substack{M,W \to \infty \\ M>W}} \hat{P}(E_M) = p_k \text{ (in probability)}$$

In this case the error probability is estimated correctly.

In the second case, T_{W_M} occurs, and so all that we can say is that

$$p_0 \leq \lim_{\substack{M,W \to \infty \\ M>W}} \hat{P}(E_M) \leq p_k$$

because the number of interfering users and thus the error probability is changing in an unknown fashion in W_M. In this case, we have no guarantees that the error probability will be estimated correctly.

We know (using the Union bound) that

$$P(T_{W_M}) \leq k \frac{W}{M} \frac{2}{q} (1 - \frac{1}{q})$$

and if we let $W = \log M$, then

$$\lim_{M \to \infty} P(T_{W_M}) \leq \lim_{M \to \infty} k \frac{\log M}{M} \cdot \frac{2}{q} (1 - \frac{1}{q}) = 0$$

Thus, the probability that we estimate the error probability of the M^{th} symbol correctly (the first case above) approaches one as $M \to \infty$ with $W = \log M$. This means that our decoder can estimate the symbol error probability when decoding the M^{th} row. Thus the capacity of this row is the same as a decoder which is given this knowledge.

When $W = \log M$ the window size becomes an insignificant fraction of the hop length M as $M \to \infty$. This means that all the rows from $\log(M) + 1$ to M will have full state information available, and will thus have full capacity. Thus the capacity of the overall scheme approaches that shown here as $M \to \infty, W = \log M$.

4. Estimating Symbol Confidence

The probability of error for each symbol at the demodulator output given the estimated state is used as the "channel measurement information" in a Chase decoding algorithm to decode each row. We must thus be able to calculate this probability from the estimated state. In this section we calculate the probability of the i^{th} symbol in a hop being in error given the state of the decodings in rows 1 to $i-1$ for the CIE channel. A symbol within a hop is considered hit when any part of it overlaps in time with another user's transmission on the same frequency.

In this development we will only calculate the effects of one interfering user. This is the dominate case in a slow-frequency-hopped system with small to moderate interference. The above can be generalized to an arbitrary number of users.

E_i = the event that the hard-detected value of the i^{th} demodulated symbol in a hop is in error (this is before decoding).

D_i = the event that an error is detected at the i^{th} symbol in a hop. This means that the demodulator and decoder estimates differ for this symbol.

T_i = the event that the interfering user's hop starts or ends during the i^{th} symbol of the reference user's hop. This is strictly a timing relation, it does not imply that any interference occurs.

H_i = the event that any part of the i^{th} symbol in a hop is hit by interference.

H_N^i = the event that T_i occurs and neither of the interfering user's hop frequencies (the frequencies before and after its hop transition) are equal to the reference user's current frequency.

H_B^i = the event that T_i occurs and both of the interfering user's hop frequencies are equal to the reference user's current frequency.

H_L^i = the event that T_i occurs and the previous (left) hop frequency of the interfering user is equal to the reference user's current frequency, but the new (right) interfering user's frequency is not equal to the reference user's current frequency.

H_R^i = the event that T_i occurs and the previous (left) hop frequency of the interfering user is not equal to the reference user's current frequency, but the new (right) interfering user's frequency is equal to the reference user's current frequency.

Note that: $P(H_N^i|T_i) + P(H_B^i|T_i) + P(H_L^i|T_i) + P(H_R^i|T_i) = 1$.

$\mathbf{S}^i$ = a ternary state vector of length $i-1$ whose elements represent the decoder knowledge about each symbol position in a hop before symbol number i. Each position, $1 \le k \le i-1$, is either: detected as an error $\mathbf{S}_k^i = d$, detected as a non-error $\mathbf{S}_k^i = \bar{d}$, or came from a row in which a row-decoder failure occurred and its state unknown $\mathbf{S}_k^i = u$.

We will be interested in estimating the reliability of a symbol after demodulation but before decoding. We can write

$$\begin{aligned} P(E_i|\mathbf{S}^i = \mathrm{s}) \;=\; \frac{1}{MP(\mathbf{S}^i = \mathrm{s})} \sum_{k=1}^{M} [\, & (1-\frac{1}{q})^2 P(E_i, \mathbf{S}^i = \mathrm{s}|H_N^k) + \\ & \frac{1}{q}(1-\frac{1}{q})\Big(P(E_i, \mathbf{S}^i = \mathrm{s}|H_L^k) + P(E_i, \mathbf{S}^i = \mathrm{s}|H_R^k)\Big) + \end{aligned}$$

$$\frac{1}{q^2}P(E_i, \mathbf{S}^i = \mathrm{s}|H_B^k)\,]$$

where the terms $P(E_i, \mathbf{S}^i = \mathrm{s}|H_X^k)$ for $X \in \{N, B, L, R\}$ are difficult to determine in general, but for the CIE channel they can be calculated as shown below.

Let $W_L(z, \mathbf{S}^i, k)$ = the cardinality of the set $\{j : \mathbf{S}_j^i = z, 1 \leq j \leq k\}$ for $1 \leq k < i$. The argument z represents one of the three state values for each symbol in the hop (u, d, or $\overline{d}$).

$W_R(z, \mathbf{S}^i, k)$ = the cardinality of the set $\{j : \mathbf{S}_j^i = z, k < j < i\}$.

$W_T(z, \mathbf{S}^i)$ = the cardinality of the set $\{j : \mathbf{S}_j^i = z, 1 \leq j < i\}$.

Under the current assumptions, we can define $P(E_i, \mathbf{S}^i = \mathrm{s}) = P(\tilde{\mathbf{S}}^i = \tilde{\mathrm{s}})$ where $\mathbf{S}_k^i = \tilde{\mathbf{S}}_k^i$ for $1 \leq k \leq i-1$ and $\tilde{\mathbf{S}}_i^i = d$ to account for E_i occurring. This unifies the treatment below so that E_i can be handled using $\tilde{\mathbf{S}}^i$. Using the CIE channel

$$\begin{aligned}
P(\tilde{\mathbf{S}}^i = \tilde{\mathrm{s}}|H_N^k) &= P(D_i|\overline{H_i})^{W_T(d,\tilde{\mathrm{s}})}(1 - P(D_i|\overline{H_i}))^{W_T(\overline{d},\tilde{\mathrm{s}})} \\
P(\tilde{\mathbf{S}}^i = \tilde{\mathrm{s}}|H_B^k) &= P(D_i|H_i)^{W_T(d,\tilde{\mathrm{s}})}(1 - P(D_i|H_i))^{W_T(\overline{d},\tilde{\mathrm{s}})} \\
P(\tilde{\mathbf{S}}^i = \tilde{\mathrm{s}}|H_L^k) &= P(D_i|H_i)^{W_L(d,\tilde{\mathrm{s}},k)}(1 - P(D_i|H_i))^{W_L(\overline{d},\tilde{\mathrm{s}},k)} \times \\
&\quad P(D_i|\overline{H_i})^{W_R(d,\tilde{\mathrm{s}},k)}(1 - P(D_i|\overline{H_i}))^{W_R(\overline{d},\tilde{\mathrm{s}},k)} \\
P(\tilde{\mathbf{S}}^i = \tilde{\mathrm{s}}|H_R^k) &= P(D_i|\overline{H_i})^{W_L(d,\tilde{\mathrm{s}},k-1)}(1 - P(D_i|\overline{H_i}))^{W_L(\overline{d},\tilde{\mathrm{s}},k-1)} \times \\
&\quad P(D_i|H_i)^{W_R(d,\tilde{\mathrm{s}},k-1)}(1 - P(D_i|H_i))^{W_R(\overline{d},\tilde{\mathrm{s}},k-1)}
\end{aligned}$$

Under the current assumptions $P(D_i|H_i) = p_1$, and $P(D_i|\overline{H_i}) = p_0$.

5. DQPSK Simulation Results

The new decoding scheme is compared to a baseline system which uses interleaving via the same frame structure as the new scheme, but decodes the rows of the frame independently. The new scheme and the baseline will be compared based on the code rate required for a fixed row-codeword error probability. It will again be assumed in the following results that the state is never incorrect.

The baseline system will have an error rate equal to the error rate of the first row-decoder in the new system. This occurs because the first row decoding in the new system has no channel state information. Thus, the actual comparison being investigated here is between the first row-decoder and the successive row-decoders (which have an increasingly large state and thus possibly better performance) in the new scheme.

In all of the following, DQPSK is the modulation format, there are 11 symbols per hop and one interfering user, background noise is neglected, the number of hop frequencies is chosen so the symbol hit probability is approximately 0.1, the Reed-Solomon code block length is 15, and Chase algorithm 3 is used.

Using a Reed-Solomon block length of 15 implies there will be two DQPSK symbols (four bits) per code symbol. In a hop, the first DQPSK symbol is a phase reference, and then the next five pairs of DQPSK symbols comprise five code symbols. This means that the largest possible state is 8 DQPSK symbols. In fact, 6 is the maximum used here (the last two row-decoders use the same size state).

Both channel cases will be used. In the CIE model, $p_0 = 0.0$ and $p_1 = 0.75$. This models the worst case interference, because an equal power interferer fully hitting both symbol slots used to make a DQPSK symbol decision only results in an error probability of 0.39. Using $p_1 = 0.75$ models the bounding behavior and also models the near/far environment. The EPI model provides a more typical environment, and thus characterizes the average behavior of the system.

Two different means of using the channel state to calculate the symbol confidence are compared. Both of these only use a window of size W on the past state knowledge rather than the entire state back to the first row in the frame. In the first, the full ternary state (error, no-error, or failure for the past decodings) is used, and in the second only the number of detected errors in the window is used. The second form of state use has a much lower complexity than the first.

For each of the four cases analyzed, two figures are shown. The first shows the codeword error probability (P_{ce}) as a function of code rate for the various rows of the frame. In this figure, the curves shown are for the first through the fifth rows of a frame. The curve for the first row corresponds to the performance of the baseline system. In the second figure, the percent improvement in code rate over the baseline system (first row) is shown versus code rate. This figure was generated from the first figure by drawing a horizontal line (fixed P_{ce}) at each of the row-one code rates. The percent code rate improvement of the second through fifth rows of the frame over the first row (at fixed P_{ce}) is then calculated using this line and linear interpolation.

5.1 CIE Channel Accuracy and Error Propagation

An analysis of the row-codeword error probability for the CIE channel using both state forms has been done. The analysis itself leads to little insight into this problem, but it is of use for assessing the accuracy of the CIE channel as an approximation to the EPI channel. Briefly, for the cases shown here, the approximation is reasonably good when the correct value for p_1 is used in place of 0.75.

Also, for the cases shown here, error propagation was shown by Monte Carlo simulation not to be a significant detriment to the overall error probability of the scheme.

5.2 CIE Channel

The figures described previously are shown in Figures 1-4 for both forms of state tracking on the CIE channel with one interferer. These come from a Monte Carlo simulation of the decoder using 100,000 frames. The probability of demodulator symbol error given hit is 0.75, and given no hit is 0.0.

These figures show that a moderate gain in code rate is possible at typical code rates, and that a sizable gain in code rate is available for low code rates. Also, since the different row curves are close together, a window size larger than one offers only a marginal improvement over a window size of one. It can also be seen that there is little to be gained by using the full ternary state; it is only necessary to keep track of the number of detected errors in the state window. These conclusions imply that a decoder which only remembers the state of the previous symbol in its hop performs nearly as well as any decoder of the form investigated here. Thus this decoder can provide a moderate performance advantage over the baseline system with only a small increase in complexity.

5.3 EPI Channel

Figures 5-8 are the result of a Monte Carlo simulation of 100,000 frames using the EPI channel with one interferer.

These figures show that the qualitative conclusions from the previous channel case are essentially valid for this more realistic case.

One difference between the two channel cases is that the rate gain is larger for the CIE model than the current case. This is because in the current case hits do not cause the appearance of demodulator errors as often as in the CIE channel. This makes the pattern of prior errors in the hop (the state) a less reliable estimate of hit or no hit at the current symbol (which is what is needed).

6. Conclusions

The proposed decoder has the same limiting capacity as any decoder of this form and, in actual (finite) practice, it can achieve a moderate code rate gain with reasonable complexity.

Based on the results of this work, a low complexity system with good performance could be built using a rate zero code for the first row of the frame, and a state window size of one.

Future work is planned in applying this basic structure to exploit the error correlation in the output of a Viterbi decoder for trellis codes.

References

[1] R. G. Gallager, *Information theory and reliable communication*. New York: Wiley, 1968.

[2] J. L. Massey, "Capacity, cutoff rate, and coding for a direct-detection optical channel," *IEEE Trans. Commun.*, vol. COM-29, pp. 1615–1621, November 1981.

[3] R. J. McEliece and W. E. Stark, "Channels with block interference," *IEEE Transactions on Information Theory*, vol. IT-30, pp. 44–53, January 1984.

[4] M. B. Pursley, "Frequency-hop transmission for satellite packet switching and terrestrial packet radio networks," *IEEE Transactions on Information Theory*, vol. IT-32, pp. 652–667, September 1986.

[5] M. B. Pursley, "Packet error probabilities in frequency-hop radio networks – Coping with statistical dependence and noisy side information," *IEEE Global Telecommunications Conference Record*, vol. 1, pp. 165–170, December 1986.

[6] D. Chase, "A class of algorithms for decoding block codes with channel measurement information," *IEEE Trasactions on Information Theory*, pp. 170–182, 1972.

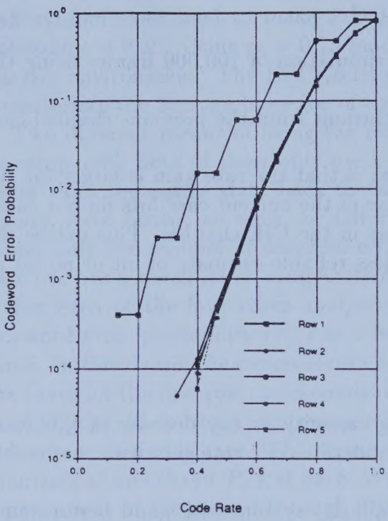


Figure 1. Codeword Error Probability vs. Rate: CIE channel, Error Count state.

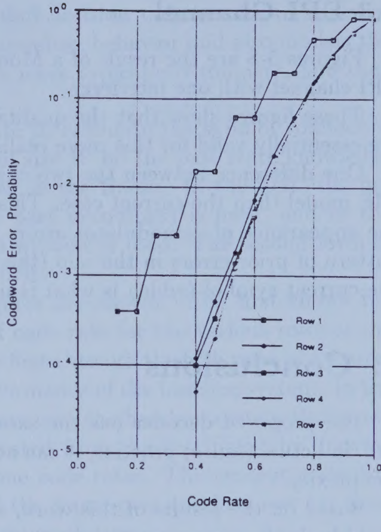


Figure 3. Codeword Error Probability vs. Rate: CIE channel, Full Ternary state.

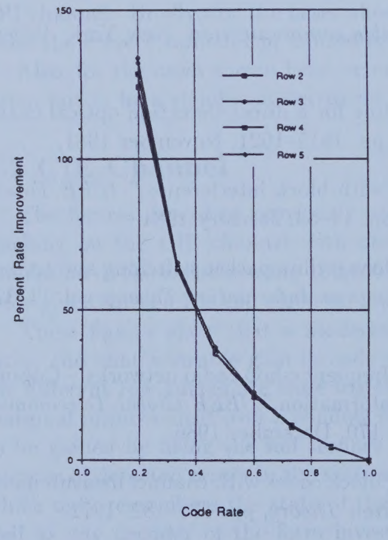


Figure 2. Percent Rate Improvement vs. Rate: CIE channel, Error Count state.

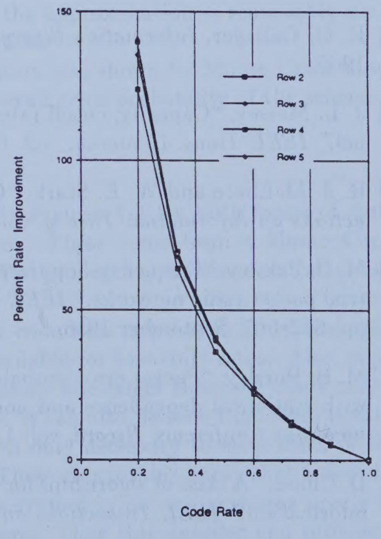


Figure 4. Percent Rate Improvement vs. Rate: CIE channel, Full Ternary state.

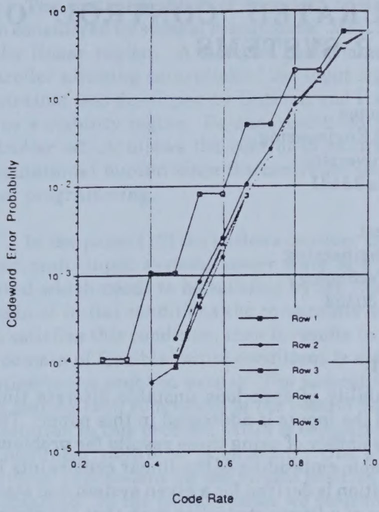


Figure 5. Codeword Error Probability vs. Rate: EPI channel. Error Count state.

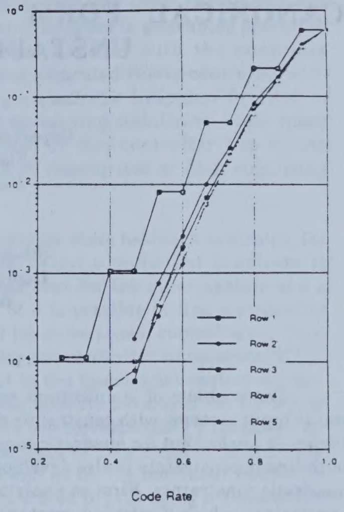


Figure 7. Codeword Error Probability vs. Rate: EPI channel. Full Ternary state.

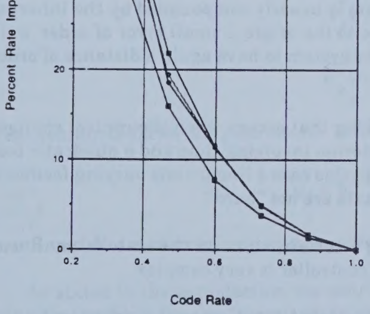


Figure 6. Percent Rate Improvement vs. Rate: EPI channel. Error Count state.

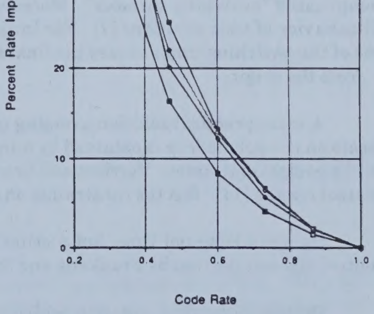


Figure 8. Percent Rate Improvement vs. Rate: EPI channel, Full Ternary state.

CANONICAL FORM OF SATURATED CONTROL OF UNSTABLE LINEAR SYSTEMS

Grazyna A. Pajunen
Department of Electrical Engineering
Florida Atlantic University
Boca Raton, Florida 33431

Erik I. Verriest
School of Electrical Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332

ABSTRACT

The problem of stabilization and null-controllability of open loop unstable discrete time multi-input systems with constraints on the states and the inputs is addressed in this paper. The theory is worked out for quadratic constraints. The feasibility of using these results for problems with linear constraints is also developed by using suitable embedding of the linear constraints in quadratic constraints. First an analytic sufficient condition is derived for a given system and state constraints, which if satisfied, guarantees the existence of a linear controller such that the closed loop system is positive invariant. Then the optimal control law is computed and the admissible set of initial conditions is given such that the input and state constraints are satisfied. Finally, the domain of feasible initial conditions is enlarged, if possible, by allowing a saturating control. The saturation is with respect to the given quadratic constraints; hence, not componentwise. These ideas are illustrated in the simulations.

1. INTRODUCTION

One practical approach to the problem of actuators saturation is the imposition of state and control limitations on the linear control. The drawback of this method is that the global stability of such systems is not guaranteed. When the plant is open loop unstable, the region from which the output can be returned to the origin is bounded. The controller must not drive the plant outside this region for, due to the state and control constraints, it may be impossible to stabilize the entire system.

As a first approach it looks reasonable to formulate the problem as an optimal control problem with constraints. In continuous time a first class of such solutions involves the minimum time controllers. These are known to be of bang-bang type, and require the storage of fairly complicated "switching surfaces". Moreover, the problem is usually compounded by the inherent ill-behavior of such solutions [7]. For initial states close to the origin a small error of order ε in one of the switching times causes the final situation of the system to have again a distance of order ε from the origin.

A more practical solution avoiding the fast switching that occurs when the control changes facets on the polyhedron is obtained by minimizing a criterion involving time and a quadratic cost on the control and states. Verriest and Lewis show that in this case a linear time varying feedback control results [11]. But the constraints on control and state are not "hard".

By optimizing not time, but a criterion involving a certain norm on the state, a nonlinear control law was derived by Frankena and Sivan [2]. The controller is very complex.

Probably the first question which needs to be answered is whether the solution to the problem exists. Can one solve a given problem using a linear or nonlinear controller? For this purpose the reachable and controllable set of states should be computed for a given system. Several procedures have been proposed in the literature to solve this problem, e.g., [3] and [5]. Most of the solutions are numerically and computationally involved.

The control problem of open loop unstable systems under state and input constraints has been considered by several researchers. Most of the systems were designed to guarantee operation in the linear region. A superior performance can be obtained however with the nonlinear controller allowing saturation of the input signal. A stabilizing saturated linear controller with constraints was developed by Gutman and Hagander [4] using a quadratic Lyapunov function to define a stability region. Pajunen and Erdol [8] proposed a time-varying stabilizing state space controller which allows the control to saturate. The limitation of this controller lies in the computational burden since the controller parameters have to be recomputed at each step using linear programming.

In the paper [12] the authors consider the design of a nonlinear state feedback controller for linear multi-input systems under state and input constraints. First a sufficient condition is derived which needs to be satisfied by the linear controller such that for the given system and a domain of initial conditions the constraints are not violated. If it is possible to find a controller that satisfies this condition, then it results in a linear optimal (in some sense) control law. Then the domain of feasible initial conditions is enlarged by allowing the controller to saturate (if the solution to the problem exists). The saturation is with respect to the (quadratic) control energy, thus scaling each component of the control equally. The largest positive invariant set for which the solution to this problem exists is computed analytically, and the corresponding expression for the nonlinear controller is obtained. One of the main advantages of this method is that the analytical expressions for the largest positive invariant set as well as for the nonlinear controller are obtained. In the present paper we reformulate the problem stated in [12] using a standard or canonical form to simplify the results and reduce the number of parameters.

The rest of the paper is organized as follows. In Section 2 the quadratic constraint problem is set up and the linear control is derived together with the domain of admissible initial conditions. In Section 3 this domain is embedded to a domain in which the control is allowed to saturate. In Section 4 the approximation to the problem with linear constraints is derived and finally an illustration by simulation is presented in Section 5.

2. PROBLEM STATEMENT AND LINEAR CONTROL DESIGN

Given the linear multi-input discrete time system

$$x_{k+1} = Ax_k + Bu_k \tag{1}$$

where $A\ \varepsilon\ R^{nxn}$ and $B\ \varepsilon\ R^{nxm}$. Find a feedback control law $u_k = Kx_k$ such that the following quadratic constraints with P and Q positive definite are satisfied for all k:

$$x_k^T P x_k \leq 1 \tag{2}$$

$$u_k^T Q u_k \leq 1 \tag{3}$$

It is convenient to factor the gain matrix K in the feedback form

$$u_k = S^{-T}\overline{D}Rx_k \ , \tag{4}$$

where S and R are respectively square roots of Q and P.

As stated in the introduction the solution to this problem was considered in [12]. Here we rewrite the problem in the "standard or canonical form" in order to reduce the number of parameters and simplify the form of the solution.

Let us introduce new variables ζ and v as follows:

$$Rx_k = \bar{\zeta}_k \tag{5}$$

$$S^T u_k = \bar{v}_k \tag{6}$$

It follows that equations (1)-(3) become respectively

$$\bar{\zeta}_{k+1} = \phi\, \bar{\zeta}_k + \Gamma\, \bar{v}_k \tag{7}$$

where $\phi = RAR^{-1}$ and $\Gamma = RBS^{-1}$

$$\|\bar{\zeta}_k\|^2 \leq 1 \tag{8}$$

$$\|\bar{v}_k\|^2 \leq 1 \tag{9}$$

Consider now the singular value decomposition of Γ

$$\Gamma = UMV^T \tag{10}$$

Let us then perform another transformation

$$\zeta_k = U^T \bar{\zeta}_k \tag{11}$$

$$v_k = V^T \bar{v}_k \tag{12}$$

The following system and constraints equations result:

$$\zeta_{k+1} = N\zeta_k + Mv_k \tag{13}$$

$$\|\zeta_k\|^2 \leq 1 \tag{14}$$

$$\|v_k\|^2 \leq 1 \tag{15}$$

In (13) the matrices N and M are:

$$N = U^T \phi U$$

$$M = U^T \Gamma \tilde{V} = \begin{matrix} \sigma_1 & \\ & \sigma_m \\ 0_{n-m} & \end{matrix} = \begin{matrix} \Sigma \\ 0_{n-m} \end{matrix}$$

Using these new variables the control law can be expressed as follows:

$$v_k = D\zeta_k \tag{16}$$

where $D = V^{-1} D U$.

In the case when the constraint (3) is strictly satisfied for all t, the closed loop system satisfies the following recursive equation.

$$\zeta_{k+1} = (N + MD)\, \zeta_k \tag{17}$$

One now tries to find the matrix D so that the sets $\psi_p = \{\zeta \mid \|\zeta\|^2 \leq p\}$ are positively invariant. From Eq. (17) it follows that a sufficient condition for constraint (14) to be satisfied is that it holds for $k = 0$ and that

$$\|N + MD\| \leq 1 \tag{18}$$

The norm is the induced Euclidean norm, i.e.,

$$\|\phi\| = \sup_{|x|=1} \|\phi x\| = \left[\lambda_{max} (\phi^T \phi) \right]^{1/2}$$

If (18) can be satisfied for some D, then the set of initial conditions satisfying (14), i.e., the set Π_1 is positive invariant, and so only the constraint (15) remains for which it is again sufficient that

$$\|v_k\| \le \|D\| \|\zeta_k\| \le \|D\| \le 1 \tag{19}$$

It is noted that if $\|\zeta_k\|$ is effectively smaller than 1, then $\|D\|$ can be allowed to exceed 1, thus leading to the possibility of a time-varying control law with the potential of speeding up the transient.

Thus our problem stated by Eq. (1)-(4) reduces to the following one: Given matrices $M \varepsilon R^{n \times m}$ and $N \varepsilon R^{n \times n}$, find a matrix $D \varepsilon R^{m \times n}$ such that

$$\|N + MD\| \le 1 \tag{20}$$

$$\|D\| \le 1 \tag{21}$$

where the norms are again the induced Euclidean norms.
Consider the following decomposition of matrix N:

$$N = \begin{bmatrix} N_1 \\ N_2 \end{bmatrix}$$

where N_1 consists of the first m rows and N_2 the last $(n-m)$ rows of the matrix N. It was shown in [12] that the minimal (Frobenius) norm of $N + MD$ is obtained for

$$D_o = -(M^T M)^{-1} M^T N = -\Sigma^{-1} N_1 \tag{22}$$

In this section we posed the problem of finding a matrix D such that the closed loop renders the sets $\psi_p = \{\zeta | \zeta^T \zeta \le p\}$ for $0 < p \le 1$ positive invariant. For this it is sufficient that D makes $\|N + MD\| \le 1$. The sufficient conditions for positive invariance are stated in the following theorem.

Theorem 1: Consider the control problem defined by Eqs. (13) and (16) and the matrix $\bar{N}_2$ consisting of the last n-m rows of N. If $\|N_2\|_F \le 1$, then there exists a feedback matrix D making the sets ψ_p positive invariant for any $p \varepsilon R^+$. If in addition $\|D\| \le 1$, then the set ψ_1 (for $p = 1$) is enclosed in the set $U_o(D) = \{\zeta | \zeta^T D^T D \zeta \le 1\}$, i.e., the linear feedback never saturates the control for the initial conditions in ψ_1.
Proof: See [12]
Remark: In the case when the system has the same number of states and inputs $(n = m)$, there always exists a feedback matrix D making the sets ψ_p positive invariant. However, when $n \ne m$, the invariance property for a given system depends on the shape of the ellipsoid defined by Eq. (2). Constraints on the input limit the admissible distance of the initial conditions from the origin.
Corollary: Under the conditions of Theorem 1 a linear feedback exists so that for every initial condition satisfying $x_o^T P x_o \le 1$ the system (1)-(4) is null controllable and the control never saturates.

Remark: Requiring the Frobenius norm of D to be less than one is quite conservative, and it may be advantageous to work directly with the quadratic forms and their associated domains $U_o(D)$ in the state space. For instance, conditioned on the spectrum of $D^T D$, we have the following cases:

(1) max $\lambda(D^TD) \le 1$, which is sufficient to guarantee that ψ_1 is embedded in $U_0(D)$.
(ii) min $\lambda(D^TD) \ge 1$, implying $U_0 \subset \psi_1$, and
(iii) min $\lambda(D^TD) < 1 < \max \lambda(D^TD)$ in which case the sets $U_0(D)$ and ψ_1 have nonempty symmetric difference.

In cases (ii) and (iii) for some initial conditions in the feasible domain, the control constraints are violated unless saturation can occur.

3. QUADRATICALLY SATURATED CONTROL

In the previous section we have shown that if $\|N_2\| \le 1$ then the linear control law (16) with $D = Do$ given by (22) when applied to the system (13) guarantees that the closed loop system is globally positively invariant and if in addition $\|D\| < 1$ the control signal will never saturate. This can happen if the set ψ_1 is totally enclosed in the set U_0 (see Fig. 1), i.e., all initial conditions satisfying constraints (2) also imply constraints (3). In the cases where the sets U_0 and ψ_1, overlap or U_0 is contained in ψ_1 not all points in ψ_1 are admissible initial conditions if one restricts the controller to be linear. Thus only local results are possible to obtain. The set of admissible initial conditions for the system (13) with linear control law (16) is the set $\psi_\theta = \{\zeta | \zeta^T\zeta \le \theta \le 1\}$. This set ψ_θ is inscribed in U_0 and is conform to ψ_1. (see Fig. 1).

Lemma 1 Given a cylinder B and an ellipsoid Ξ centered about the origin as follows:

$$B : y^T Y y = 1$$

$$\Xi : y^T Z y = 1$$

The smallest ellipsoid Ξ_1 that is conform to Ξ and just inside B is given by

$$y^T Z y = = \lambda^{-1}_{max}\left(Z^{-1}Y\right)$$

where λ_{max} is the maximum eigenvalue.

Proof: Since Ξ_1 is conform to Ξ, it is of the form

$$y^T Z y = q \qquad for\, some\ q > 0.$$

At the point of tangency we must have

$$y^T Y y = 1 \qquad y \varepsilon B$$

$$y^T Z y = 1 \qquad y \varepsilon \Xi$$

and

$$Yy = kZy \qquad for\, some\ k$$

Multiplying the latter by y^T on the left and substituting, results

$$k = \frac{1}{q}$$

This means that $Yq - Z$ is singular. Since the smallest ellipsoid corresponds to the smallest possible q, we find

$$q = \lambda^{-1}_{max}(Z^{-1}Y)$$

QED

Applying this lemma to our problem we have

$$\Xi = I, \ \mathrm{B} = D_o^{\ T} D_o, \quad thus\ q = \lambda_{max}^{-1} (D_o^{\ T} D_o)$$

QED

In summary, if $\|N_2\| \leq 1$ then the linear controller (16) with $D = D_0$ can be used to control the system (13). When $\theta = \lambda_{max}^{-1} (D_0^T D_0) = 1$ then all initial conditions satisfying constraints (2) also satisfy constraints (3). When, however, $\theta < 1$, the set of admissible initial conditions is $\psi_\theta = \{\zeta | \zeta^T \zeta \leq \theta\}$ and $\psi_\theta \subset \psi_1$.

In order to increase the admissible set of initial conditions, a nonlinear (saturating) control law is proposed. To avoid discontinuities in the control function u(t), the saturating control should be consistent with the criterion. This leads one to define the quadratically saturating feedback control as follows:

$$\mathrm{v}_k^c = \begin{cases} \mathrm{v}_k & if\ \mathrm{v}_k^T \mathrm{v}_k \leq 1 \\ \dfrac{\mathrm{v}_k}{\left(\mathrm{v}_k^T \mathrm{v}_k\right)^{1/2}} & if\ \mathrm{v}_k^T \mathrm{v}_k > 1 \end{cases} \tag{23}$$

where v_k is the control that would be obtained if the domain of the linear feedback controller is extended to the whole state space. Thus v_k is computed using Eq. (16) with $D = D_0$. Note that in the control law (23) we do not saturate the components as in the linear case, but each component of the vector v_k is scaled equally. The set of admissible initial conditions for the system (13) with the nonlinear control law (23) is the set $\psi_\alpha = \{\zeta | \zeta^T \zeta \leq \alpha_0\}$ with $0 < \theta \leq \alpha_0 \leq 1$.

Lemma 2: If

$$N_2 \neq 0 \ \ and \ \ \|N_2\|_F < 1 \ \ then \ \ J = \|N + MD/\alpha\|_F^2 \leq 1 \quad for$$

$$\alpha \,\varepsilon \left[1, \alpha_o\right] \quad where \ \alpha_o = \|N_1\|_F / \left(\|N_1\|_F - \sqrt{1 - \|N_2\|_F^2} \right)$$

and the matrices N_1 and N_2 consist of first m rows and last n-m rows of matrix N, respectively.
Proof: $\|N_2\| \leq 1$ is the condition for positive invariance. In the case when $J \neq 0$ if $\|N_2\| < 1$ there is a possibility for improvement. Since $J_0 = \|N + MD_0\|_F < 1$ introducing a saturating controller which is equivalent to a time-varying controller D_0/α where $\alpha \,\varepsilon\, [1, \alpha_0]$, the improvement of the performance can be obtained. α_0 can be computed as follows

$$J_o = \|N + MD_o / \alpha_o\|_F^2 = 1$$

$$J_o = \|N - \frac{1}{\alpha_o} M(M^T M)^{-1} M^T N\|_F^2 = \| \left[I - \frac{1}{\alpha_o} M(M^T M)^{-1} M^T \right] N\|_F^2$$

Then

$$J_o = \left\| \begin{matrix} (1 - \frac{1}{\alpha_o}) N_1 \\ N_2 \end{matrix} \right\|_F^2 = 1$$

thus the result follows. QED

Remark: Based on Lemma 2 we can see that the increase of admissible set of initial conditions can be obtained if the Frobenius norm of N_2 is strictly smaller than one ($\|N_2\|_F < 1$). The smaller $\|N_2\|_F^2$ the greater increase is obtained.

Theorem 2: The sets $\psi_\rho = \{\{\zeta | \zeta^T \zeta \leq \rho \ \varepsilon \ R^+\}$ are positive invariant for all $D(\alpha) = D/\alpha$ where $1 \leq \alpha \leq \alpha_0$ and $\alpha_0 = \|N_1\| / (\|N_1\| - \sqrt{1 - \|N_2\|^2})$.
The proof of this theorem is direct application of Lemma 1.
Corollary: Let U_1 be set

$$\left\{\zeta_o \mid \zeta_o^T D^T D \zeta_o \le \alpha_o^2\right\}$$

and ψ_2 be the largest set ψ_p $(p \le 1)$ enclosed in it (i.e., $\psi_\alpha = \{\zeta | \zeta^T \zeta \le q\}$ where $q = \alpha_0^2 \lambda_{max}^{-1}(D_o^T D_o)$). All points of ψ_α are null controllable by the quadratic saturating feedback.

Proof: By Theorem 2 the sets ψ_p are positive invariant for all $D(\alpha)$. If $p \le 1$ then the state constraints are satisfied and the positive invariance is assured. The set U_1 is the set for which the input constraints are satisfied but there may exist some points in U_1 for which the state constraints are not satisfied. Thus, the largest set ψ_p, $p \le 1$ inscribed in U_1 and which is conformed to ψ_1 and ψ_θ is the set of initial conditions for which all constraints are satisfied and the system remains positive invariant and can be computed using Lemma 2.

QED

Remark: Once the state enters the positive invariant set ψ_θ saturation can no longer occur. Once v_k is computed the problem can be inverted to compute the actual control signal as

$$u_k = \begin{cases} D_1 x_k & \text{if } u_k^T Q u_k \le 1 \\ \dfrac{D_1 x_k}{\left(u_k^T Q u_k\right)^{1/2}} & \text{if } u_k^T Q u_k > 1 \end{cases}$$

where $D_1 = (S - {}^T V)^{-1} D_0 U^T R$ and the admissible initial conditions are $x^T P x \le q$.

4. LINEAR CONSTRAINTS

In many problems of practical importance the constraints on the states and the controls are linear. A direct attak on the linear constraint problem is very complex so here we propose an alternative approach by relating the linear constraint problem to a quadratic constraint problem and then using a solution involving a quadratic saturation.

Assume first that there are exactly n linear state constraints "in general position", i.e., non-redundant or nontrivial, say

$$|c_i x|, \le 1 \quad c_i \in R^n \quad ; \quad i = 1, \dots, n \tag{24}$$

or in compact form (24) is with $G = [\, c_i^T, c_2^T, \dots, c_n^T]^T$ and $X_o = [\, x_1, \dots, x_n\,]$,

$$G X_0 = \Omega_o \tag{25}$$

The idea is to replace (24), which is a parallellopipeded, symmetrical about the origin, by the ellipsoid of largest volume that is inscribed in it. In [12] we have proven the following:

Lemma 3: The ellipsoid of maximal volume inscribed in the parallellopipeded $|Gx| \le 1$ is given by the surface $E(\,\Omega_o = I)$

$$x^T G^T G x = 1 \tag{26}$$

Now solve problem of Section 2 but with P replaced by $G^T G$ for the linear constraint problem. Similarly m linear constraints $|h_i^T u| \le 1$ on the m-dimensional input can be replaced by a quadratic constraint $u^T H^T H u \le 1$ in the input space. The solution is straightforward, but of course this is at the expense of some conservatism.

The following procedure was derived in [12]:

(i) Make a choice of n of the q constraints, giving $G_{[n]}$.

(ii) Evaluate μ_i (as function of Ω_0) for the remaining q-n constraints, $i=n+1,...,q$.

(iii) Check the existence of a Ω_0 with diag $\Omega_0 = I$, for which $\mu_i < 1$; $i=n+1,...,q$, while also not compromising on the 'volume' of the ellipsoid. Recall that a maximum volume is obtained for $\Omega_0 = I$.

(iv) Combinatorial search: Check the above for all choices of n constraints from the $q>n$ given ones.

It is clear that a lot of heuristics may be used in order to curtail the combinatorial explosion. Or trial and error methods can be used until a 'reasonable' design is found in this multiple constraint problem.

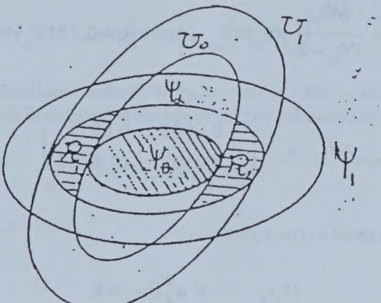


Figure 1. Null-controllability region.

ψ_1: Constraint region for the states. ψ_0: Region of null-controllability with linear control. ψ_a: Region of null-controllability with quadratic saturating control. U_o: Constraint region of the linear feedback control. U_1: Constraint region of the quadratic saturating feedback control.

5. SIMULATION RESULTS

The system considered is given as follows:

$$x_{k+1} = \begin{bmatrix} 0.8 & 0.5 \\ -0.4 & 1.2 \end{bmatrix} x_k + \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} x_k. \tag{27}$$

$$x_k^T \begin{bmatrix} 0.0016 & 0.0001 \\ 0.0001 & 0.0016 \end{bmatrix} x_k \leq 1 \tag{28}$$

$$u_k^T \begin{bmatrix} 0.02 & 0 \\ 0 & 0.02 \end{bmatrix} u_k \leq 1 \tag{29}$$

The eigenvalues of the matrix A are $\lambda_{1,2} = 1 \pm j\,0.4$. Rewriting this sytem in a canonical form (13)-(15) yields

$$N = \begin{bmatrix} 1.05 & -0.692 \\ 0.235 & 0.95 \end{bmatrix} ; M = \begin{bmatrix} 0.2915 & 0 \\ 0 & 0.2739 \end{bmatrix}$$

$$U = \begin{bmatrix} 0.7071 & 0.7071 \\ 0.7071 & 0.7071 \end{bmatrix} ; V = \begin{bmatrix} 0.7071 & 0.7071 \\ 0.7071 & 0.7071 \end{bmatrix}$$

Since $n=m$, the matrix $N_2 = 0$ so ignoring constraints on the input we are always able to find D such that the feedback system is positive invariant.

In this case we find

$$D_o = \begin{bmatrix} -3.6015 & 2.3735 \\ -0.8575 & -3.4689 \end{bmatrix}$$

Since $\|D_o\| = 4.65 > 1$, no all initial conditions satisfying (72) are admissible. The admissible set of initial conditions with this linear control is:

$$\psi_\theta = \left\{ \zeta \mid \zeta^T \zeta \leq \theta \right\} \quad where\ \theta = \lambda_{max}^{-1}\left(D_o^T D_o\right) = 0.0463$$

Using the saturating controller proposed in this paper the set of admissible initial conditions can be extended to $\psi_2 = \{\zeta \mid \zeta^T \zeta \leq q\}$ and q is computed as follows:

$$\alpha_o = \frac{\|N\|_F}{\|N\|_F - 1} = 3.975 \quad ; q = \alpha_o^2 \theta = 0.7315$$

and

$$D_{min} = D_o / \alpha_o = \begin{bmatrix} -0.9060 & 0.5971 \\ -0.2157 & -0.8727 \end{bmatrix}$$

and $\|D_{min}\| = 1.17 > 1$.

Transforming the original system yields

$$u_k = \begin{cases} D_1 x_k & if\ u_k^T Q u_k \leq 1 \\ \dfrac{D_1 x_k}{(u_k^T Q u_k)^{1/2}} & otherwise \end{cases}$$

with

$$D_1 = \begin{bmatrix} 0.4 & -1.2 \\ -0.80 & -0.5 \end{bmatrix}$$

and the set of admissible initial conditions is

$$x_o^T P x_o \leq 0.7315$$

The simulation results using the saturating controller and the initial conditions $x_o^T = [15 \quad 14]$ are shown in Figure 2.

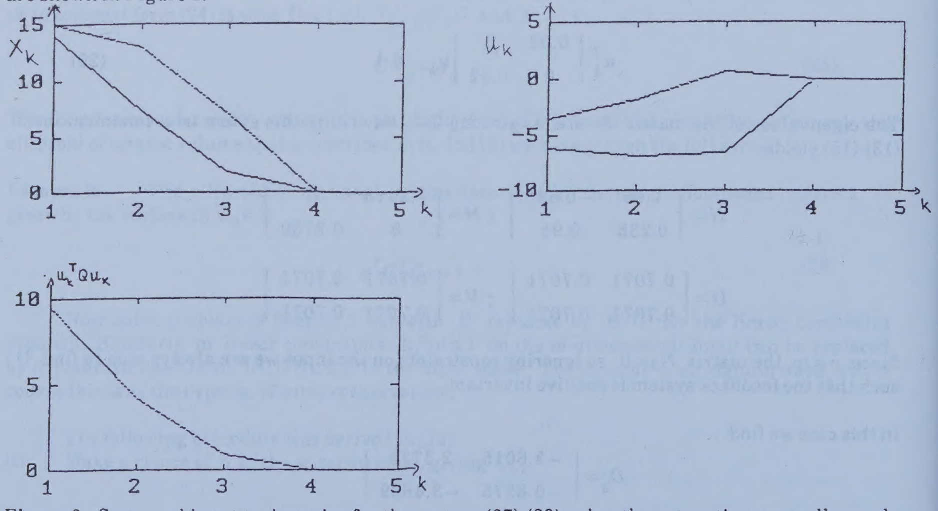


Figure 2. State and input trajectories for the system (27)-(29) using the saturating controller and the initial conditons $x_o^T = [15 \quad 14]$.

Conclusions

In this paper we considered the problem of stabilization and null-controllability of open-loop unstable multi-input systems with state and input constraints. First, ignoring input constraints, the sufficient condition for a given system to be positive invariant was derived. We have shown that when the number of states is equal to the number of inputs, this condition is always satisfied. Otherwise, it depends on the shape of the ellipsoid describing the quadratic state constraints. The constraint on the input limits the size of this ellipsoid or the set of admissible initial conditions. We have shown that the maximum size of this set is

$$x_o^T P x_o \le \theta\,, \quad where\ \theta = \left[\lambda_{max}\left(D_o^T D_o\right)\right]^{-1}$$

and Do is the linear state feedback controller gain matrix. We have shown also that by using a nonlinear (saturating) control law the admissible set of initial conditions can be extend to

$$x_o^T P x_o \le \alpha_o^2 \theta\ ;\ where\ \alpha_o \ge 1$$

and can be computed using Lemma 2.

References

[1] R. F. Brammer, "Controllability in Linear Autonomous Systems with Positive Controllers", SIAM Journal Control, Vol. 10, No. 2, pp. 339-353, 1972.

[2] J. F. Frankena and R. Sivan, "A Non-linear Optimal Control Law for Linear Systems", Int. J. Control, Vol. 30, No. 1, pp. 159-178, 1979.

[3] P. O. Gutman and M. Cwikel, "An Algorithm to Find Maximal State Constrained Sets for Discrete-Time Dynamical Systems with Bounded Controls and States', IEEE Transactions Automatic Control, Vol. 32, No. 3, pp. 251-254, 1987.

[4] P. D. Gutman and P. Hagander, "A New Design of Constrained Controllers for Nonlinear Systems", IEEE Transactions Automatic Control, Vol. 30, No. 1, pp. 22-33, 1985.

[5] J. B. Lasserre, "On Reachable and Controllable Sets for Two-Dimensional Linear Discrete-Time Systems, (to appear) Journal of Optimization Theory Applications.

[6] E. Lee and L. Markus, *Foundations of Optimal Control Theory*, John Wiley, New York, 1967.

[7] G. J. Olsder, "Time-Optimal Control of Multivariable Systems Near the Origin", Journal of Optimization Theory Applications, Vol. 16, Nos. 5/6, pp. 497-517, 1975.

[8] G. A. Pajunen and N. Erdol, "Time-Varying Saturating Feedback Control of Linear Systems Under State and Control Constraints", Proceedings 29th IEEE Conference on Deceision and Control, Honolulu, Hawaii, pp. 3644-4645, 1990.

[9] L. Pandolfi, "Linear Control Systems: Controllability with Constrained Controls", Journal of Optimization Theory Applications, Vol. 19, No. 4, pp. 577-585, 1976.

[10] M. Vassilaki, J. F. Hennet, and G. Bitsoris, "Feedback Control of Linear Discrete-time Systems under State and Control Constraints", International Journal Control, Vol. 47, No. 6, pp. 1717-1735, 1988.

[11] E. I. Verriest and F. L. Lewis, "On the Linear Quadratic Minimum-Time Problem", (to appear) IEEE Transactions Automatic Control, Vol. 36, No. 6, July 1991.

[12] E. I. Verriest and G. A. Pajunen, "Saturated Control of Unstable Open Loop Systems", Proceedings of the 30th IEEE Conference on Decision and Control and submitted to IEEE Transactions Automatic Control.

VARIABLE STRUCTURE OUTPUT FEEDBACK CONTROL: SWITCHING SURFACE DESIGN

Reyad El-Khazali and Raymond A. DeCarlo
School of Electrical Engineering
Purdue University
West Lafayette, IN, 47907

ABSTRACT

This paper describes a procedure for variable structure output feedback control (VSOFC) for a class of multivariable linear time-invariant systems. The equivalent control method of Utkin in the output feedback mode yields a reduced order system exhibiting an output feedback equivalent dynamic. Using the Kimura-Davison sufficient condition for pole-placement using output feedback, a necessary and sufficient condition is derived to assign the eigenvalues of the "reduced" order system. The observability and controllability of the reduced system is discussed. A design algorithm is introduced based on the system structure in the regular form.

1. INTRODUCTION

Variable structure systems (VSS) are systems in which the control is a high speed switched control law. The main advantage of applying VSS to nonlinear systems are due to the reduction of order of the original plant, disturbance rejection, good robustness, and possible linearization through the control [1-8].

The application of VSS linear and nonlinear systems have been limited to full-state feedback, which requires extra dynamics to measure the unavailable states [7]. This increases the complexity of the design procedure, and, in some cases, duplicates the size of the controlled system. Consequently, this has led to use the output feedback approach. (See, for example, [16-21].)

The problem of using output feedback in VSS has been investigated by several authors. Among these, the stable, robust tracking by sliding mode control [12] has been developed for open loop stable systems with equal numbers of inputs and also outputs, and also for the case when the so-called equivalent control is not unique. On the other hand, the stabilizability of linear systems using VSS with output feedback has been introduced in [9]. However, the observability of the *reduced order* system (which is necessary for pole assignability and stabilizability) was not addressed. The problem was also addressed in [10]. The control scheme used was a combination of a static output feedback using any existing algorithm [17] in conjunction with a switched control. The switched control is used mainly to relocate the eigenvalues of the switching surface dynamics.

In this work we wish to consider the same problem and address all the points associated with the complete assignability to the reduced system. We also discuss the observability of the reduced order system and introduce a switching surface design algorithm. The control is a high speed switched control in terms of the output variables.

The design procedure of variable structure control (VSC) system is a two phase process. The first phase is to choose a set of switching surfaces such that the original system restricted to the intersection of the switching surfaces has a desired behavior. The second phase is to determine a switched control law which forces the system's trajectory to and maintains it on the sliding surface [1-4].

The switching surface design in variable structure output feedback control (VSOFC) is introduced based on assigning a desired set of eigenvalues to the reduced order system. Necessary and sufficient conditions for pole-placement using output feedback [16-20] are used to derive a necessary and sufficient condition to design a switching surface map.

One type of switching surface in VSOFC uses the static output variables, $y(t)$, and has the form

$$\sigma(y) = Sy(t) \tag{1.1}$$

where $y(t) \in \mathbf{R}^r$ is the output vector, and $S \in \mathbf{R}^{m \times r}$ is the switching surface matrix.

The advantage of using VSOFC is to avoid using an observer to estimate the unavailable states and yet still achieve the desired control objective.

The difficulties of using such a method are: i) The equivalent system trajectory in the sliding mode is projected onto a lower dimensional subspace where part of it is fixed and equals to the null space of the output matrix C, ii) the equivalent control is not unique when the product CB is not full column rank, and iii) the observability of the reduced order system is not guaranteed and has to be met. This follows because the reduced dynamics exhibit an equivalent output feedback structure where observability is necessary for pole placement using output feedback [9,10,12]. Such constraints limit the class of systems that can be considered.

The purpose of this paper is to investigate the first phase of the design of VSOFC for a class of linear time-invariant systems. For simplicity, the discussion of this paper concentrate on systems linear in the control assuming ideal switching (infinitely fast) in the control law [1].

The paper is organized as follows: Problem statement and system definition are introduced in section II. Section III introduces some preliminaries and background including the observability of the reduced order system. The switching surface design is examined in IV.

2. SYSTEM DEFINITION AND PROBLEM STATEMENT

In general, we wish to consider a linear time-invariant system described by the equation

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \tag{2.1}$$

where $x \in \mathbf{R}^n$, $u \in \mathbf{R}^m$, $y \in \mathbf{R}^r$ are the state, control, and output vectors, respectively; A, B, and C are constant maps of appropriate dimensions

Assumption 1. The system is assumed to be completely controllable and completely observable, rank$(B) = m$, rank$(C) = r$, and Kimura-Davison condition is satisfied, i.e., $n \leq m + r - 1$.

In variable structure output feedback control [1-8], the control vector component, u_i, of u, takes two values,

$$u_i(y) = \begin{cases} u_i^+(y) & when \quad \sigma(y) > 0 \\ u_i^-(y) & when \quad \sigma(y) < 0 \end{cases} \tag{2.2}$$

depending on the location of the system's output representative point, y(t), with respect to the switching manifold $\sigma = 0$.

The switching surface or manifold in VSOFC takes the form

$$\sigma(y) = [\ \sigma_1(y), \ \cdots, \ \sigma_m(y)\ ]^t = Sy(t) \tag{2.3}$$

where S is an $m \times r$ real parameter matrix to be designed. This type of switching surface was introduced in [9,10].

The switching surface, $\sigma = Sy$, in the output space $(\mathbf{R}^r)$ is chosen to stabilize the system in the sliding mode by driving the output vector to the switching surface whose dynamics are predetermined. Thus the design process breaks down into two phases.

First Phase: Given a linear time-invariant system (2.1) and a set of n-m distinct numbers $\Lambda = \{\lambda_1, \ \cdots, \lambda_{n-m}\}$. The objective is to find a real $m \times r$ map, S, such that n-m eigenvalues of (2.1) in the sliding mode are precisely those of the set Λ.

The first phase implies a reduction of order of (2.1) to an (n-m)-dimensional equivalent system. This is because the n-dimensional state dynamics of (2.1) must satisfy the m algebraic equations, $\sigma(y) = 0$. Such constraints reduce the equivalent system from an n^{th} order system to an $(n-m)^{th}$ order system [1,2,8].

Second Phase: The second phase of design is to determine a switched control of the form (2.2) such that the system's trajectory is directed toward the sliding surface from any point in the output space. This condition corresponds to global output stability to the switching surface [1].

3. PRELIMINARIES AND BACKGROUND

The method of equivalent control [1] may be used to introduce the structure of the system restricted to the switching surface. If the system is initially on the switching surface, then $\sigma(y(t_o)) = 0$, and $\dot{\sigma}(y(t_o)) = 0$; therefore

$$\dot{\sigma} = SCAx(t) + SCBu(t) = 0 \tag{3.1}$$

This results in the equivalent control law [9,10,12]

$$u_{eq}(t) = -(SCB)^{-1} SCAx(t) \tag{3.2}$$

which exists uniquely if (SCB) is nonsingular. Substituting (3.2) into (2.1) yields the equivalent system

$$\dot{x}(t) = [(\ I_n - B(SCB)^{-1}SC\ )\ A]x(t) \tag{3.3a}$$

$$y(t) = Cx(t) \tag{3.3b}$$

Equation (3.3) represents the system dynamics in the sliding mode. The goal is to choose the function $\sigma = Sy(t)$ provided that (SCB) is invertible, and a switched output feedback control of the form (2.2) to adjust the state dynamics of (3.3) to achieve the desired behavior of the output trajectory.

Clearly, the equivalent control is not unique when CB is rank deficient. The case when CB is not full rank, the number of outputs equals to the number of inputs, (i.e., $r = m$), and when the open loop system is stable is discussed in [12]. In our case, the open loop system need not be stable.

To design a switching surface of the form (2.3) one needs to investigate the equivalent system (3.3a). The following results show the importance of the number of inputs with respect to the number of outputs.

Lemma 3.1. Let $r = m$, CB be full rank, $S \in \mathbf{R}^{m \times r}$ be a nonsingular map such that (SCB) is invertible, and let $\sigma = Sy(t)$. The equivalent system of (3.3a) is independent of S.

Proof. The sliding surface in the output space is specified by $Sy(t) = 0$, i.e., it is the null space of S. Since S is nonsingular, then the sliding surface and the system dynamics of (3.3a) are invariant regardless of the choice of S [9]. In other words, the motion of the state trajectory in the sliding mode is determined by the projector $[I_n - B(SCB)^{-1}SC]$ which projects $\mathbf{R}^n$ onto an (n-m)-dimensional subspace equal to $N[SC]$, where $N[SC]$ is the null space of SC [13]. But, when $m = r$, $N[SC] = N[C]$ which is invariant. Hence, the result follows. □

Notice that when $r < m$ the equivalent control is not unique. If CB has full column rank then $r \geq m$, i.e., the number of outputs is at least equal to the number of inputs. Hence, the nonsingularity of SCB depends exclusively on the choice of the switching surface matrix, S. If S is chosen to have full row rank and $N[S] \bigcap Im[CB]$, then SCB is nonsingular, where $Im[CB]$ denotes the range space of the columns of the matrix CB. This results in a uniquely defined equivalent control. Thus, the following assumption becomes necessary.

Assumption 2. CB has full column rank.

Lemma 3.2. Let $r > m$, CB have full column rank, $S \in \mathbf{R}^{m \times r}$ be a full row rank map such that SCB is nonsingular, and let $\sigma = Sy(t)$, then the equivalent system of (3.3a) is dependent on S.

Proof: When $r > m$, $N[C] \subset N[SC]$. The state dynamics given by (3.3a) in the sliding mode are projected onto $N[SC]$. The null space of SC is spanned by n-m n-vectors, n-r of which are determined by $N[C]$. i.e., one may write $N[SC] = N[C] \oplus \mathscr{S}$, where $\mathscr{S}$ is an (r-m)-dimensional subspace depends on the choice of S □.

Lemmas 3.1 and 3.2 show the necessity of having more outputs than inputs. To justify further assumptions, and to have more insight to the problem, the essential features of our problem become clear when viewed in the appropriate coordinate system. One may transfer system (2.1) into the *regular form* [14]. The system dynamics in the regular form is given by

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} 0 \\ B_2 \end{bmatrix} u(t)$$

$$y = \begin{bmatrix} C_1 & C_2 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \tag{3.5}$$

where $B_2 \in \mathbf{R}^{m \times m}$ is nonsingular. The switching surface in the new coordinate is:

$$\sigma(z_1, z_2) = S \begin{bmatrix} C_1 & C_2 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = [0] \tag{3.6}$$

where $z_1 \in \mathbf{R}^{n-m}$, $z_2 \in \mathbf{R}^{m}$, A_{11}, A_{12}, A_{21}, A_{22}, C_1, and C_2 are real matrices of appropriate dimensions.

Remark 3.1. By assumption 2, CB is full rank. The regular form of C and B implies that $CB = C_2 B_2$. Since B_2 is an $m \times m$ nonsingular matrix, $\text{rank}[C_2] = m$, i.e. C_2 has full rank.

Reduced Order Dynamics

Suppose that the system's output representative point is initially on the switching surface, then $\sigma(z) = 0$ and $\dot{\sigma}(z) = 0$. Equation (3.6) then yields,

$$z_2 = -(SC_2)^{-1} S C_1 z_1 \tag{3.7}$$

and the reduced order dynamics are given by

$$\dot{z}_1 = (A_{11} - A_{12}(SC_2)^{-1} S C_1) z_1 \doteq (A_{11} - A_{12} K C_1) z_1 \tag{3.8}$$

where $K \doteq (SC_2)^{-1} S$ represents an output feedback gain map; A_{12}, and C_1 are the input and output matrices of the reduced order system, respectively. Clearly, S must be chosen such that SC_2 is nonsingular.

Switching surface design reduces to the construction of S so that the reduced order system, equation (3.8), has a desired behavior. If the objective is to assign a prescribed set of eigenvalues to the reduced order system, then the problem is to find S so that equation (3.8) has a desired spectrum, Λ, i.e., $\sigma(A_{11} - A_{12} K C_1) = \Lambda$ where $K = (SC_2)^{-1} S$.

Equation (3.8) exhibits an output feedback structure in K. Sufficient conditions exists for a solution to the spectral assignment problem [16,17]. However, under what condition does there exist an S so that the spectrum is assigned? This constrains the class of K's. The next two propositions characterize the class of permissible K matrices.

Proposition 3.1. Let $K \in \mathbf{R}^{m\times r}$, $C_2 \in \mathbf{R}^{r\times m}$ be full rank of rank m, $r > m$, and let $S \in \mathbf{R}^{m\times r}$ be chosen such that (SC_2) is nonsingular, then a necessary and sufficient condition for there to exist a solution, S, of the equation

$$K = (SC_2)^{-1}S \tag{3.9}$$

is:

$$KC_2 = I_m \tag{3.10}$$

Proof.(if) Multiplying (3.9) by C_2 on right side gives,

$$(SC_2)^{-1}SC_2 = KC_2 = I_m$$

(only if) Suppose there exists $K \in \mathbf{R}^{m\times r}$ such that $KC_2 = I_m$, show that there exists S such that $(SC_2)^{-1}S = K$. Let $S = K$, then

$$(SC_2)^{-1}S = (KC_2)^{-1}K = I_m\ K = K \qquad \text{Q.E.D}$$

If a solution to (3.9) exists, it must satisfy (3.10) for which K is a left inverse of C_2, and the switching surface design process is equivalent to finding a gain feedback K that will assign a set $\Lambda = \{ \lambda_1, \ldots, \lambda_{n-m} \}$ to the reduced order system (3.8) subject to the constraint given by (3.10). From the proof of proposition 3.1, one may choose a desired switching surface, $Sy = 0$, by letting $S = K$.

Proposition 3.2. Let $C_2 \in \mathbf{R}^{r\times m}$ have rank m, $r > m$, and let $K \in \mathbf{R}^{m\times r}$ be such that $KC_2 = I_m$, then the set $\mathbf{K}_1$ of all solutions of (3.10) is nonempty and is given by

$$\mathbf{K}_1 = \left\{ K \mid K = C_2^{-L} + \Gamma M^t \right\} \tag{3.11}$$

where $\Gamma \in \mathbf{R}^{m\times(r-m)}$ is an arbitrary parameter matrix, and where the rows of $M^t \in \mathbf{R}^{(r-m)\times r}$ span the left annihilator of C_2. i.e., $M^tC_2 = [0]$.

Substituting the form of K given by (3.11) into (3.8) yields,

$$\dot{z}_1 = (\hat{A}_{11} - A_{12}\,\Gamma\,\hat{C}_1)z_1 \tag{3.12}$$

where $\hat{A}_{11} \doteq (A_{11} - A_{12}C_2^{-L}C_1)$, and $\hat{C}_1 \doteq M^tC_1$.

The constraints imposed on $\mathbf{K}_1$ destroy m^2 free parameters. The question now is what freedom is left to assign the proper eigenvalues to the reduced order system. This requires that the submatrices of the reduced order system, A_{12}, and $\hat{C}_1$ have certain structural properties. This will be investigated in the next section.

Observability of the Reduced Order System

Equation (3.12) defines the reduced order system in terms of a new output matrix, $\hat{C}_1$, and a new system matrix $\hat{A}_{11}$. The controllability and the observability of the triple $(\hat{C}_1, A_{12}, A_{11})$ is necessary to assign a desired set of eigenvalues to (3.12). Observe that the controllability of the pair (A, B) guarantees the controllability of the pair (A_{11}, A_{12}) [6]. Since $\hat{A}_{11}$ can be seen as a state feedback modification of A_{11} through the input matrix A_{12}, then by lemma 2.1 [14] the pair $(\hat{A}_{11}, A_{12})$ is also controllable.

Since the controllability of (3.12) is guaranteed, a sufficient condition for pole placement by static output feedback requires observability and the satisfaction of Davison-Kimura condition [16,18]. Unfortunately, observability of the reduced system is not as clear as controllability. The following examples illustrate the cases of observability for both the original system (2.1) and the reduced order system (3.12).

Example 3.1. Consider the linear time-invariant system of (2.1) in the regular form, where

$$A = \begin{bmatrix} 1 & 0 & -2 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ -1 & 0 & 2 & 0 & 1 \\ 0 & 1 & 1 & 1 & 2 \\ -1 & 1 & -1 & 2 & -1 \end{bmatrix}, \; B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \; C = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

The pair $(C,\ A)$ is observable while the pair $(A_{11},\ C_1)$ is not.

Example 3.2: Consider the following linear system in the regular, where

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Here $(C,\ A)$ is not observable yet $(C_1,\ A_{11})$ is observable.

Our form of switching surface design requires that one choose Γ to assign the spectrum to (3.12). This requires observability of the pair $(\hat{C}_1,\ \hat{A}_{11})$ and controllability of $(\hat{A}_{11},\ A_{12})$, which is already guaranteed. The question is how does the observability of $(C_1,\ A_{11})$ affect the observability of $(\hat{C}_1,\ \hat{A}_{11})$.

If $(C_1,\ A_{11})$ is observable, then $(C_1,\ \hat{A}_{11})$ is also observable because $C_2^{-L} C_1$ can be viewed as a simple output feedback map which does not affect observability. Further if $Im[C_1] \bigcap Im[C_2] = \phi$, then the pair $(\hat{C}_1,\ \hat{A}_{11})$ is observable. This is of course a sufficient condition. The following example indicates that if $Im[C_1] \bigcap Im[C_2] \neq \phi$, then the pair $(\hat{C}_1,\ \hat{A}_{11})$ is not necessarily observable.

Example 3.3. Consider the linear time-invariant system (2.1) in the regular form with

$$A = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \; B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \; C = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

The system is already in the regular form. In this case, $\hat{C}_1 = M^t C_1 = [\ 0\ 1\ ]$, and

$$\hat{A}_{11} = A_{11} - A_{12}C_2^{-L}C_1 = \begin{bmatrix} -1 & 1 \\ 0 & 0 \end{bmatrix}$$

Clearly, $(C,\ A)$ and $(C_1,\ A_{11})$ are both observable while $(\hat{C}_1,\ \hat{A}_{11})$ is not.

If the pair $(\hat{C}_1, \hat{A}_{11})$ is not observable but is detectable, then it is possible to show that the reduced order dynamics are stabilizable [15].

4. SWITCHING SURFACE DESIGN

Switching surface design is equivalent to determining a desired reduced order dynamics in the sliding mode. Since (3.12) exhibits an output feedback equivalent structure, the theory of pole-placement using output feedback [16-21] can be used, among other theories, to set up an appropriate output feedback map. A sufficient condition to assign the eigenvalues of the reduced order system is derived in terms of the ranks of the input and output matrices [16].

Lemma 4.1 [17]: Consider the linear system (2.1). If the triple (A, B, C) is complete, if B and C have full column and row rank respectively, and if $n \leq m + r - 1$, then the system is always pole-assignable using output gain feedback provided that a slight modification of the poles to be assigned can be tolerated.

Lemma 4.1 asserts the existence of a proper output feedback map that relocates the eigenvalues of (2.1) to desired locations. The goal here is to apply this result to (3.12). It is obvious that the ranks of A_{12}, and C_1 play important rules in determining the number of eigenvalues that can be assigned to (3.12).

Recall that $\hat{A}_{11} \in \mathbf{R}^{(n-m)\times(n-m)}$, $A_{12} \in \mathbf{R}^{(n-m)\times m}$, $\hat{C}_1 \in \mathbb{R}^{(r-m)\times(n-m)}$, and that $n \leq m + r - 1$ (i.e., the original system satisfies the Davison-Kimura condition) with $m \geq 2$ and $r > m$.

Theorem 4.1. Suppose the pair $(\hat{C}_1,\ \hat{A}_{11})$ is observable. Let $\Lambda = \{\lambda_1, \cdots, \lambda_{n-m}\}$ be a symmetric set of $n - m$ complex numbers. If $rank[A_{12}] = m$ (full column rank) or $rank[A_{12}] = (n-m)$ (full row rank), then there exists a real matrix $\Gamma \in \mathbf{R}^{m\times r-m}$ such that $\sigma(\hat{A}_{11} - A_{12}\Gamma\hat{C}_1) = \Lambda$, or at least within an arbitrary small neighborhood of Λ.

Proof. First the pair $(\hat{A}_{11},\ A_{12})$ is controllable by [7]. Hence the triple $(\hat{C}_1,\ \hat{A}_{11},\ A_{12})$ is complete. The result follows from lemma 4.1 provided that

$$n - m \leq rank[A_{12}] + rank[\hat{C}_1] - 1 \tag{4.1}$$

Since $M^tC_2 = [0]$, and $rank[C_1\ C_2] = r > m$, $rank[\hat{C}_1] = rank[M^tC_1] = r - m$. Since the original system satisfies the Davison-Kimura condition,

$$n - m \leq r - 1 = m + (r - m) - 1$$

If $rank[A_{12}] = m$, equation (4.1) is satisfied.

On the other hand if $rank[A_{12}] = n - m$, then

$$n - m \leq (n - m) + (r - m) - 1$$

Since $r - m > 1$. Again, equation 4.1 is satisfied. □

Once Γ is found using any existing output feedback algorithm [17,20,21], the switching surface map is easily found by letting $S = K = C_2^{-L} + \Gamma M^t$. Other choices of S are possible. The following proposition characterizes the set of all possible S's associated with a given K.

Proposition 4.1. Let $K \in \mathbf{K}_1$ and $rank[S] = m$. A switching surface matrix S satisfies $(SC)^{-1}S = K$ if and only if

$$S[I_r - C_2K] = [0] \tag{4.2}$$

Proof. If (4.2) is true and if $rank[S] = $ m, then $N[S] = Im[I_r - C_2K]$. But $[I_r - C_2K]$ is a projector of rank $r - m$, by virtue of the fact that $KC_2 = I_m$. Hence

$$N[S] = Im[I_r - C_2K] = N[K]$$

Since $N[K] \bigcap Im[C_2] = \phi$ because $KC_2 = I_m$,

$$N[S] \bigcap Im[C_2] = \phi$$

Hence SC_2 is nonsingular. As such, equation 4.2 implies that $S = SC_2K$ or equivalently $(SC_2)^{-1}S = K$.

Conversely if $K = (SC_2)^{-1}S$ then $SC_2K = S$ or equivalently $S[I_r - C_2K] = [0]$.

Theorem 4.1 states sufficient conditions for the existence of a switching surface matrix, S, to achieve a desired spectrum of the reduced order system, i.e. the original system is the sliding mode. Proposition 4.1 characterizes the relationship between admissible feedbacks $K = C_2^{-L} + \Gamma M^t$ and allowable switching surface matrices, S, which will result in the desired spectrum. Extra freedom in the choice of S can be used to achieve other design constraints.

REFERENCES

[1] Utkin *Sliding Modes and Their Application in Variable Structure Systems,* MIR Publishers, Moscow, 1978.

[2] R.A. DeCarlo, S.H. Żak, and G.P. Matthews, "Variable structure control of nonlinear multivariable systems: A tutorial," *IEEE Proc.*, Vol. 76, pp. 212-232, March 1988.

[3] V.I. Utkin, "Variable structure systems with sliding modes", *IEEE Trans. Automat. Control,* Vol. AC-22, No. 2, pp. 212-222, 1977.

[4] V.I. Utkin, "Variable Structure Systems," *Automatica,* No. 9, pp. 5-25, Sept., 1983.

[5] V.I. Utkin and K.D. Yang, "Methods for Constructing Discontinuity Planes in Multidimensional Variable Structure Systems," *Automatica,* No. 10, pp. 72-77, Oct., 1978.

[6] K.K.D. Young, P.V. Kokotovic, and V.I. Utkin, "A singular perturbation analysis of high-gain feedback systems," *IEEE Trans. Aut. Control,* Vol. AC-22, No. 2, pp. 931-938, 1977.

[7] A.G. Bondarev, S.A. Bondarev, N.E. Kostyleva, and V.I. Utkin, "Sliding Modes in Systems with Asymptotic state Observers," *Automatica,* No. 6, pp. 6-11, June, 1985.

[8] U. Itkis, "*Control Systems of Variable Structure,*" J. Wiley and Sons Inc, New York, 1976.

[9] B.S. Heck and A. A. Ferri, "Application of output feedback to variable structure systems," *J. Guidance,* Vol. 12, No. 6, pp. 932-935.

[10] B.A. White, "Applications of output feedback in variable structure control," Royal Military college of Science, Chap, 8. pp. 145-169.

[11] S.M. Madani-Esfahani, R.A. DeCarlo, M.J. Corless, and S.H. Żak, "On Deterministic Control of Uncertain Nonlinear Systems," *Proc. American Contr. Conf.* (Seattle, WA), pp. 1523-1528, June, 1986.

[12] G.C. Verghese, B. Fernandez R., and J.K. Hedrick, "Stable, robust tracking by sliding mode Control," *System & Control letter,* No. 10, pp. 27-34, 1988.

[13] O.M.E. El-Ghezawi, A.S. Zinober, and S.A. Billings " Analysis and Design of Variable Structure Systems Using a Geometric Approach," *Int. J. Control,* Vol. 38, No. 6, pp. 1121-1134, 1983.

[14] A.G. Luk'yanov and V.I. Utkin, "Methods of reducing equations of dynamic systems to regular form," *Automat. Remote Control,* No. 4, pp 5-13, 1981.

[15] W.M. Wonham, *Linear Multivariable Control*: A Geometric Approach," Third Edition, Springer-Verlag, New York, 1979.

[16] H. Kimura, "Pole Assignment by Gain Output Feedback," *IEEE Trans. Automat. Control,* Vol. 20, No. 4, pp. 509-516, 1975.

[17] H. Kimura, "A further Result on the Problem of Pole Assignment by Output Feedback," *IEEE Trans. Automat. Control,* Vol. 22, No. 3, pp. 458-463, 1977.

[18] E.J. Davison, "On Pole Assignment in Linear Systems with Incomplete State Feedback," *IEEE Trans. Automat. Control,* Vol. AC-15, No. 3, pp. 348-351, 1970.

[19] E.J. Davison, and R. Chatterjee, "A note on Pole Assignment in Linear Systems with Incomplete State Feedback," *IEEE Trans. Automat. Control,* Vol. AC-16, pp. 98-99, Feb., 1971.

[20] E.J. Davison, and S.H. Wang, "On Pole Assignment in Linear Multivariable Systems Using Output Feedback," *IEEE, Trans. Automat. Control,* pp. 516-518, 1975.

[21] A.N. Andry, JR., E.Y. Shapiro, J.C. Chung, "Eigenstructure Assignment for Linear Systems," *IEEE Trans. Aerospace and Electronic Systems,* Vol. AES-19, No. 5, pp. 711-728, September, 1983.

REDUCED-ORDER SOLUTION FOR A CLASS OF LINEAR QUADRATIC OPTIMAL CONTROL PROBLEMS

D. Skataric[1)], Z. Gajic[2)], and Dj. Petkovski[3)]

[1)]Energoproject, Lenjinov Bulevar 12, 11070 N. Beograd, Yugoslavia.

[2)]Rutgers University, Department of Electrical and Computer Engineering, Piscataway, NJ 08855–0909, USA.

[3)]University of Novi Sad, Faculty of Technical Sciences, V. Vlahovica 3, 21000 Novi Sad, Yugoslavia.

Abstract

The reduced-order solution is obtained for a class of linear-quadratic optimal control problems having weakly interconnected system matrix, strongly connected control matrix, and with a special structure for the state penalty matrix. An example demonstrates the effectiveness of the proposed reduced-order algorithm. The presented method is very well suited for parallel implementation.

I. Problem Formulation

Consider a linear dynamical system given by

$$\dot{x} = Ax + Bu \tag{1}$$

with

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad A = \begin{bmatrix} A_1 & \varepsilon A_2 \\ \varepsilon A_3 & A_4 \end{bmatrix}, \quad B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \tag{2}$$

where $x_i \in R^{n_i}$, i = 1, 2 are state vectors, $u \in R^m$ is a control input and ϵ is a small parameter. A quadratic type functional to be minimized is associated with (1) in the form

$$J = \int_0^\infty \left[x^T Q x + u^T R u \right] dt, \quad Q \geq 0, \ R > 0 \tag{3}$$

All matrices defined in (1)-(3) are constant and of appropriate dimensions.

The system matrix A, defined in (2) has the structure of the weakly coupled systems [1]-[2]. However, due to strongly coupled control matrix B this system does not belong to the class of weakly coupled linear control systems. In this paper we will show that despite the strong coupling coming from the input matrix, the order-reduction can be achieved, like in the case of purely weakly coupled systems, by using the specific structure for the state penalty matrix.

Up to authors best knowledge the problem order-reduction through the choice of the state penalty matrix Q has not been studied in the control literature. Thus, the engineering relevance of this paper is to study the linear-quadratic optimal control problem [3] of (1)-(3), in the spirit of parallel and distributed reduced-order algorithms [4], under the following assumption.

Assumption 1. The state penalty matrix Q has the structure

$$Q = \begin{bmatrix} Q_1 & \varepsilon Q_2 \\ \varepsilon Q_2 & \varepsilon Q_3 \end{bmatrix} \tag{4}$$

This choice of the matrix Q is quite common in engineering practice since the control engineers hardly penalize all state variables by weighting factors of the same magnitude, especially for large scale systems.

II. Main Result

The optimal problem of minimizing (3) along trajectories of (1) has a very well known solution given by [3]

$$u_{opt} = -F_{opt}x = -R^{-1}B^TPx \tag{5}$$

where P is the positive semi-definite stabilizing solution of the algebraic Riccati equation

$$A^TP + PA + Q - PSP = 0, \quad S = BR^{-1}B^T \tag{6}$$

In the following we will show that the nature of the solution of (6) subject to the partition of the problem matrices defined in (2)-(4) is given by

$$P = \begin{bmatrix} P_1 & \varepsilon P_2 \\ \varepsilon P_2^T & \varepsilon P_3 \end{bmatrix} \tag{7}$$

and then, derive the reduced-order algorithm for finding P.

Partitioning (6) compatible to (2), (4) and (7) we get three algebraic equations

$$\begin{aligned} &P_1A_1 + A_1^TP_1 + Q_1 - P_1S_1P_1 + \varepsilon^2\left(P_2A_3 + A_3^TP_2^T\right) - \\ &\varepsilon\left(P_1S_2P_2^T + P_2S_2^TP_1 + \varepsilon P_2S_3P_2^T\right) = 0 \end{aligned} \tag{8}$$

$$\begin{aligned} &P_1A_2 + P_2A_4 + A_1^TP_2 - P_1S_1P_2 - P_1S_2P_3 + Q_2 + \\ &\varepsilon\left(A_3^TP_3 - P_2S_2^TP_2 - P_2S_3P_3\right) = 0 \end{aligned} \tag{9}$$

$$\begin{gathered} P_3 A_4 + A_4^T P_3 + Q_3 + \\ \varepsilon\left(P_2^T A_2 + A_2^T P_2 - P_2^T S_1 P_2 - P_2^T S_2 P_3 - P_3 S_2^T P_2 - P_3 S_3 P_3\right) = 0 \end{gathered} \tag{10}$$

where

$$S_1 = B_1 R^{-1} B_1^T, \quad S_2 = B_1 R^{-1} B_2^T, \quad S_3 = B_2 R^{-1} B_2^T \tag{11}$$

Since ϵ is a small parameter we can define an $O(\epsilon)^1$ approximation of (8)-(10) as follows

$$P_1^{(0)} A_1 + A_1^T P_1^{(0)} + Q_1 - P_1^{(0)} S_1 P_1^{(0)} = 0 \tag{12}$$

$$P_1^{(0)} A_2 + P_2^{(0)} A_4 + A_1^T P_2^{(0)} - P_1^{(0)} S_1 P_2^{(0)} - P_1^{(0)} S_2 P_3^{(0)} + Q_2 = 0 \tag{13}$$

$$P_3^{(0)} A_4 + A_4^T P_3^{(0)} + Q_3 = 0 \tag{14}$$

The unique positive semi-definite stabilizing solution for P, obtained from (12)-(14), and defined by

$$P^{(0)} = \begin{bmatrix} P_1^{(0)} & \varepsilon P_2^{(0)} \\ \varepsilon P_2^{(0)^T} & \varepsilon P_3^{(0)} \end{bmatrix} \tag{15}$$

exists under the following assumption.

Assumption 2. The triple $\left(A_1, B_1, \sqrt{Q_1}\right)$ is stabilizable-detectable and the matrix A_4 is stable.

Since all solutions obtained from (12)-(14) are O(1), it can be concluded that our staring assumption (7) about the nature of the solution of (6) is correct.

From (12)-(14) we have obtained the first order approximation of the required solution in terms of the completely decomposed reduced-order algebraic equations.

In the next step we will derive the reduced-order parallel algorithm, based on the fixed point iterations [4], for obtaining the solution of P up to any arbitrary degree of accuracy.

Defining the approximation errors as

$$P_j = P_j^{(0)} + \varepsilon E_j, \quad j = 1, 2, 3 \tag{16}$$

and using (15) in (8)–(10) and (12)-(14), we get the following error equations

$$\begin{gathered} E_1 D_1 + D_1^T E_1 = \varepsilon E_1 S_1 E_1 - \varepsilon\left(P_2 A_3 + A_3^T P_2^T\right) + \\ P_1 S_2 P_2^T + P_2 S_2^T P_1 + \varepsilon P_2 S_3 P_2^T \end{gathered} \tag{17}$$

---

[1] $O(\epsilon^k)$ stands for $C\epsilon^k$, where C is a bounded constant and k is any arbitrary constant.

$$
\begin{aligned}
E_2 A_4 + D_1^T E_2 = \varepsilon \left(E_1 S_1 E_2 + E_1 S_2 E_3 \right) - E_1 D_2 \\
+ P_1^{(0)} S_2 E_3 - A_3^T P_3 + P_2 S_2^T P_2 + P_2 S_3 P_3
\end{aligned}
\tag{18}
$$

$$
E_3 A_4 + A_4^T E_3 = P_2^T S_1 P_2 + P_2^T S_2 P_3 + P_3 S_2^T P_2 + P_3 S_3 P_3 - P_2^T A_2 - A_2^T P_2 \tag{19}
$$

where

$$
D_1 = A_1 - S_1 P_1^{(0)}, \quad D_2 = A_2 - S_1 P_2^{(0)} - S_2 P_3^{(0)} \tag{20}
$$

Let us propose the following reduced-order parallel algorithm for solving (17)-(19)

$$
\begin{aligned}
E_1^{(i+1)} D_1 + D_1^T E_1^{(i+1)} = \varepsilon E_1^{(i)} S_1 E_1^{(i)} - \varepsilon \left(P_2^{(i)} A_3 + A_3^T P_2^{(i)^T} \right) + \\
P_1^{(i)} S_2 P_2^{(i)^T} + P_2^{(i)} S_2^T P_1^{(i)} + \varepsilon P_2^{(i)} S_3 P_2^{(i)^T}
\end{aligned}
\tag{21}
$$

$$
\begin{aligned}
E_2^{(i+1)} A_4 + D_1^T E_2^{(i+1)} = \varepsilon \left(E_1^{(i+1)} S_1 E_2^{(i)} + E_1^{(i+1)} S_2 E_3^{(i+1)} \right) - E_1^{(i+1)} D_2 \\
+ P_1^{(0)} S_2 E_3^{(i+1)} - A_3^T P_3^{(i+1)} + P_2^{(i)} S_2^T P_2^{(i)} + P_2^{(i)} S_3 P_3^{(i+1)}
\end{aligned}
\tag{22}
$$

$$
\begin{aligned}
E_3^{(i+1)} A_4 + A_4^T E_3^{(i+1)} = P_2^{(i)^T} S_1 P_2^{(i)} + P_2^{(i)^T} S_2 P_3^{(i)} + P_3^{(i)} S_2^T P_2^{(i)} + \\
P_3^{(i)} S_3 P_3^{(i)} - P_2^{(i)^T} A_2 - A_2^T P_2^{(i)}
\end{aligned}
\tag{23}
$$

with initial conditions

$$
E_1^{(0)} = 0, \quad E_2^{(0)} = 0, \quad E_3^{(0)} = 0 \tag{24}
$$

The following theorem indicates the features of the proposed algorithm (21)-(24).

Theorem 1. Under conditions stated in Assumption 2, the algorithm (21)-(24) converges to the exact solution of the error term, and thus to the required solution P, with the rate of convergence of $O(\epsilon)$, that is

$$
\left\| E_j - E_j^{(i+1)} \right\| = O\left(\varepsilon\right) \left\| E_j - E_j^{(i)} \right\| \tag{25}
$$

or equivalently

$$
\left\| E_j - E_j^{(i)} \right\| = O\left(\varepsilon^{i+1}\right), \quad i = 0, 1, 2, ..; \quad j = 1, 2, 3 \tag{26}
$$

Proof. As a starting point, we need to show the existence of a bounded solution of (17)-(19) in the neighborhood of $\epsilon = 0$. By the implicit function theorem [5], it is enough to show that the corresponding Jacobian is nonsingular at $\epsilon = 0$. The Jacobian at $\epsilon = 0$ is given by

$$J(\varepsilon)_{|\varepsilon=0} = \begin{bmatrix} J_{11} & 0 & 0 \\ J_{21} & J_{22} & J_{23} \\ 0 & 0 & J_{33} \end{bmatrix} \tag{27}$$

where

$$\begin{aligned} J_{11} &= I_{n_1} \otimes D_1^T + D_1^T \otimes I_{n_1} \\ J_{22} &= I_{n_1} \otimes D_1^T + A_4^T \otimes I_{n_2} \\ J_{33} &= I_{n_2} \otimes A_4^T + A_4^T \otimes I_{n_2} \end{aligned} \tag{28}$$

where $\otimes$ stands for the Kronecker product representation. For the Jacobian to be nonsingular the block diagonal elements J_{ii}, i = 1, 2, 3 have to be nonsingular. The matrix D_1 is a closed loop matrix, and thus stable by a well known property of the algebraic Riccati equation and by Assumption 2. By the same assumption, the matrix A_4 is stable, so that by the property of the Kronecker product [6], matrices J_{ii} are nonsingular. Thus, for ϵ small enough, the Jacobian is nonsingular.

In the next step we have to show the convergence of the algorithm (21)-(24) and give an estimate of the rate of convergence. For i = 0, (17) and (21) imply

$$\left(E_1 - E_1^{(0)}\right) D_1 + D_1^T \left(E_1 - E_1^{(0)}\right) = \varepsilon f_1(E_1, E_2, \varepsilon) \tag{29}$$

Since D_1 is stable and E_1 and E_2 are bounded it follows that

$$\left\| E_1 - E_1^{(0)} \right\| = O(\varepsilon) \tag{30}$$

Similarly from (19) and (23) we have

$$\left(E_3 - E_3^{(0)}\right) A_4 + A_4^T \left(E_3 - E_3^{(0)}\right) = \varepsilon f_3(E_2, E_3, \varepsilon) \tag{31}$$

so that

$$\left\| E_3 - E_3^{(0)} \right\| = O(\varepsilon) \tag{32}$$

The use of the same arguments in (18) and (22) produces

$$\left\| E_2 - E_2^{(0)} \right\| = O(\varepsilon) \tag{33}$$

Continuing the same procedure and by induction we conclude that

$$\left\| E_1 - E_1^{(i)} \right\| = O\left(\varepsilon^{i+1}\right) \tag{34}$$

$$\left\| E_2 - E_2^{(i)} \right\| = O\left(\varepsilon^{i+1}\right) \tag{35}$$

$$\left\| E_3 - E_3^{(i)} \right\| = O\left(\varepsilon^{i+1}\right) \tag{36}$$

with i = 1, 2, ..., which completes the proof of Theorem 1.

It is obvious that the proposed algorithm (21)-(24) can be implemented as a synchronous one [4]. The study is underway to prove the convergence of the corresponding asynchronous algorithm [4].

The importance of the existence of the $O(\epsilon^i)$, i = 1, 2, 3,... approximations for the small parameter linear control theory is demonstrated in [7] where the accuracy of $O(\epsilon^6)$ is necessary to stabilize the considered plant-filter augmented weakly coupled system.

III. Numerical Example

The following fourth order linear-quadratic control problem example demonstrates the efficiency of the proposed method. Problem matrices are taken from [9]

$$A = \begin{bmatrix} -0.75 & 0.28125 & 0.15 & 0.31875 \\ -0.25 & -1.15625 & -0.35 & -0.24375 \\ -0.75 & 0.28125 & -3.25 & -0.28125 \\ -0.25 & -1.15625 & 1.25 & -1.84375 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$y = Cx = \begin{bmatrix} 1.5 & 1.9375 & 0.5 & 0.0625 \\ -0.25 & 0.71875 & 0.25 & 0.28125 \end{bmatrix} x$$

$$Q = C^T C, \quad R = I_2, \quad \varepsilon = 0.1$$

Simulation results are obtained by using the L-A-S package for computer aided control system design [8], and presented in Table 1. The optimal value for the criterion is J_{opt} = 1.8222.

From Table 1 we can notice very good numerical behavior of the proposed algorithm consistent with the statement of Theorem 1.

i	J(i) - Jopt
1	0.16855 x E-2
2	0.66638 x E-4
3	0.28597 x E-5
4	0.22424 x E-6
5	0.14966 x E-7
6	0.10897 x E-8
7	0.47606 x E-10
8	0.33538 x E-11
9	0.3694 x E-12

Table 1. Errors in the criterion approximation per iteration

An other example of weakly coupled systems having strong coupling through the input matrix is a binary distillation column considered in [10]. That control system can be numerically decomposed and solved in terms of the reduced-order problems by choosing the state penalty matrix according to Assumption 1.

Even more, the presented method is applicable to almost any linear control system with prescribed degree of stability [11], since in that case we are working with A + αI which is block diagonally dominant for α large enough.

IV. Conclusion

It has been shown that the linear quadratic control problem of the weakly coupled subsystems strongly coupled through the input matrix can be numerically decomposed and solved in terms of the reduced-order parallel algorithms by using a specific form for the state penalty matrix. The considered choice of the state penalty matrix is quite natural in practice since control engineers hardly penalize all state variables by the weighting factors of the same magnitude.

References

[1] P. Kokotovic, W. Perkins, J. Cruz Jr., and G. D'Ans, "ϵ—coupling for near-optimum design of large scale linear systems", Proc. IEE, Vol. 116, 889–892, 1969.

[2] Z. Gajic, Dj. Petkovski and X. Shen, *Singularly Perturbed and Weakly Coupled Linear Control Systems — A Recursive Approach.* New York: Springer Verlag, 1990.

[3] H. Kwakernaak and R. Sivan, *Linear Optimal Control Systems.* New York: Wiley, 1972.

[4] D. Bertsekas and J. Tsitsiklis, "Some aspects of parallel and distributed iterative algorithms — A survey", Automatica, Vol. 27, 3–21, 1991.

[5] J. Ortega and W. Rheinboldt, *Iterative Solution of Nonlinear Equations on Several Variables.* New York: Academic Press, 1970.

[6] P. Lancaster and M. Tismenetsky, *The Theory of Matrices.* Orlando: Academic Press, 1985.

[7] X. Shen and Z. Gajic, "Near optimum steady state regulators for stochastic linear weakly coupled systems", Automatica, Vol. 26, 919–923, 1990.

[8] P. West, S. Bingulac and W. Perkins, "L-A-S: A Computer aided control system design language", in *Computer Aided Control System Engineering.* Eds. M. Jamshidi and Herget, Amsterdam: North-Holland, 1985.

[9] L. Shien and Y. Tsay, "Transformations of a class of multivariable control systems to block companion forms", IEEE Trans. on Automatic Control, AC-27, 199–202, 1982.

[10] S. Bhattacharyya, A. Del Nero Gomes, and J. Howze, "The structure of robust disturbance rejection control", IEEE Trans. Automatic Control, AC-28, 874–881, 1983.

[11] B. Anderson and J. Moore, *Optimal Control — Linear Quadratic Methods.* Englewood Cliffs: Prentice Hall, 1990.

ROBUST ALGORITHMS FOR TRANSFORMATION OF MIMO SYSTEMS BETWEEN CONTINUOUS- AND DISCRETE-TIME DOMAINS

S. Bingulac and H. F. VanLandingham
The Bradley Department of Electrical Engineering
Virginia Polytechnic Institute and State University
Blacksburg, Virginia 24061-0111

Abstract.

New numerically robust algorithms are presented for converting linear continuous-time constant-parameter state models into equivalent discrete-time state models (*discretization*) as well as the reverse problem of determining continuous-time models to represent given discrete-time models (*continualization*). Two methods are considered for discretization: one that assumes that the inputs are held constant *(ZOH)*, and one in which the inputs are linearly interpolated between samples. The development is carried out in the time domain, working with the appropriate state space models. Using basic properties of the exponential and logarithm function, a unified approach has been developed which provides simple robust algorithms for system discretization and continualization using the two methods mentioned above.

1. Introduction

With the widespread use of computers in control loops it is inevitable that control engineers will face problems associated with sampled-data systems. Such systems by their very definition contain a mixture of continuous-time and discrete-time signals. A common problem that arises with sampled-data control systems is to find the equivalent effect of continuous-time operations as seen by the computer in the loop. Typically, the modeling of the signal converters assumes an ideal uniform sampler for the analog-to-digital converter and a simple (zero-order) hold device synchronized with the samples for a digital-to-analog converter. With these assumptions one may find in many references the standard *zero-order hold (ZOH)* model which will be discussed subsequently [1,2].

In addition to simple plant modeling with ZOH equivalents there are occasions, such as in digital redesign, that demand more accuracy between a given continuous-time system and its discrete-time equivalent model. In these instances higher-order discrete models are required. A model, [1], which has been used for greater aaccuracy than the standard ZOH assumes a linearly interpolated input. This method is a *ramp invariant* model which contrasts with the standard *ZOH* model being a *step invariant* model. We will refer to it as a *First-Order hold (FOH)*, (although classical *FOH* is defined differently). There are many other useful models, but this paper will focus on only these two methods of discretization as being the most useful in practice.

The reverse problem, called *continualization*, is that of reconstructing a continuous-time model from a given discrete-time model. This problem could arise, for instance, when measured discrete data are used to identify a continuous-time system [6]. The method of continualization is presented for each of the two discretization techniques, thereby offering the designer flexibility in going between the continuous and the discrete domains.

2. Problem Formulation

In the area of Systems and Controls, as well as related areas such as Signal Processing, it is useful to be able to discretize a given continuous-time system. We assume a basic state variable representation for a continuous-time system as follows. A *state space realization* for a linear, continuous-time, constant parameter system consists of a 4-tuple of matrices; namely,

$$R_c = \{A_c, B_c, C_c, D_c\} \tag{1}$$

which defines the state model

$$\begin{aligned} \dot{x}(t) &= A_c x(t) + B_c u(t) \\ y(t) &= C_c x(t) + D_c u(t) \end{aligned} \tag{2}$$

where *x(t), u(t)* and *y(t)* are the state, input and output vectors with dimensions *n, m* and *p*, respectively, while the matrices A_c, B_c, C_c and D_c are constant matrices with compatible dimensions.

Discretization Procedure

In this paper some computational issues of the discretization procedures will be discussed with emphasis on explaining an easily implementable algorithm.

The familiar ZOH equivalent discrete-time (DT) model assumes that the input vector *u(t)* in (2) is constant between (uniform) samples. The equivalent DT model can be represented as

$$R_d = \{A_d, B_d, C_d, D_d\} \tag{3}$$

which implies the DT state model

$$\begin{aligned} x(k+1) &= A_d x(k) + B_d u(k) \\ y(k) &= C_d x(k) + D_d u(k) \end{aligned} \tag{4}$$

The matrices A_d and B_d are related to A_c and B_c in (2) by the well known relations, [1,2,3],

$$A_d = e^{A_c T} = \sum_{i=0}^{\infty} \frac{(A_c T)^i}{i!} \tag{5}$$

and
$$B_d = \int_0^T e^{A_c t} B_c dt = \sum_{i=0}^{\infty} \frac{(A_c T)^i}{(i+1)!} B_c T \tag{6}$$

Also, if A_c is nonsingular, $$B_d = A_c^{-1}(e^{A_c T} - I) B_c \tag{7}$$

And since the output *y(t)* in (2) is assumed to be ideally sampled, the matrices $C_d = C_c$ and $D_d = D_c$. In the following, the basic discretization technique is discussed.

Algorithm: (*F-E-A$_d$*) Zero- and First-Order Hold Models This algorithm provides a robust method for the conversion from a continuous-time model (2) to a four or five matrix discrete-time state model, represented either by (3) or the following representation

$$R_{df} = \{A_d\,,\, B_{d0}\,,\, B_{d1}\,,\, C_d\,,\, D_d\} \tag{8}$$

which, in turn, can be written as

$$\begin{aligned} x(k+1) &= A_d x(k) + B_{d0} u(k) + B_{d1} u(k+1) \\ y(k) &= C_d x(k) + D_d u(k) \end{aligned} \tag{9}$$

The matrices A_d, B_d, C_d and D_d have been described in the previous section, see (5) and (6). However, a simple truncation of the series (5) or (6) is not sufficient for matrices A_c and sampling intervals T satisfying that

$$\|A_c T\| > 1 \tag{10}$$

e.g., a truncated version of (5) may either require large N, leading to considerable round-off errors, or may not converge at all, [5]. In contrast, the algorithm presented here is completely general.

It has been shown, [1], that the *ZOH* model can be calculated using an intermediate matrix E as follows:

$$A_d = I + EA_cT\,, \qquad B_d = EB_cT \tag{11}$$

where

$$E = \sum_{i=0}^{\infty} \frac{(A_cT)^i}{(i+1)!} \tag{12}$$

To specify the remaining matrices, we define

$$F = \sum_{i=0}^{\infty} \frac{(A_cT)^i}{(i+2)!} \tag{13}$$

from which we obtain

$$B_{d0} = (E - F)B_cT\,, \qquad B_{d1} = FB_cT \tag{14}$$

And, if A_c is nonsingular,

$$F = (A_d - I - A_cT)(A_cT)^{-2}$$

It is desirable to create an algorithm which allows condition (10) as well as singular A_c matrices. The development for this algorithm, referred to as Algorithm *F-E-*A_d, is based on the well known property of the exponential function

$$\exp(x) = e^x = \left(e^{(x/r)}\right)^r \tag{15}$$

By comparing power series (12) and (13), it may be determined that the matrices E and F satisfy the following relation:

$$E = I + FA_cT \tag{16}$$

It is convenient to introduce the scaling parameter j as follows:
As will be made clear later, we define

$$j = \left[\frac{\ln(\|A_c T\|)}{\ln(2)}\right]_{\substack{integer\\ part}} + 1 \tag{17}$$

Thus, using property (15) with $r = 2^j$, $\|A_c T/r\| < 1$ and the scaled series will converge.

It is easily shown that the matrix A_d given in (5) may be calculated using the following recursive process:

$$A_{dk+1} = A_{dk}^2\,, \quad for \quad k = 1, 2, \cdots, j \tag{18}$$

where

$$A_{d1} = \sum_{i=0}^{N} \frac{(A_c T/r)^i}{i!} \tag{19}$$

with j and r defined as in (17). Also, from (11) we obtain the following recursions:

$$\begin{aligned} A_{dk} &= I + E_k A_c T_k \\ A_{dk+1} &= I + E_{k+1} A_c T_{k+1} \end{aligned} \tag{20}$$

where $T_{k+1} = 2\,T_k$.

Introducing (20) into (18), we obtain the following:

$$\begin{aligned} (I + E_k A_c T_k)^2 &= I + E_{k+1} A_c T_{k+1} \\ I + 2 E_k A_c T_k + (E_k A_c T_k)^2 &= I + E_{k+1} A_c T_{k+1} \end{aligned} \tag{21}$$

leading finally to

$$\begin{aligned} T_{k+1} &= 2 T_k \\ E_{k+1} &= E_k (I + E_k A_c T_k / 2) \end{aligned} \tag{22}$$

Using (16) and (18), we can write, similarly to (20), the recursion equations

$$\begin{aligned} E_k &= I + F_k A_c T_k \\ E_{k+1} &= I + F_{k+1} A_c T_{k+1} \end{aligned} \tag{23}$$

where $T_{k+1} = 2\,T_k$.

Using now (22) and eliminating E_k and E_{k+1} from (23), the following relationship between F_k and F_{k+1} can be derived:

$$\begin{aligned} T_{k+1} &= 2\,T_k \\ F_{k+1} &= 0.5\,F_k + 0.25\,(I + F_k A_c T_k)^2 \end{aligned} \tag{24}$$

for $k = 1, 2, 3, \cdots, j$ where

$$T_1 = \frac{T}{r}\,, \qquad F_1 = \sum_{i=0}^{N} \frac{(A_c T/r)^i}{(i+2)!} \tag{25}$$

for $r = 2^j$ and
$$j = \left[\frac{\ln(\|A_c T\|)}{\ln(2)}\right]_{\substack{integer\\part}} + 1 \tag{26}$$

The desired $F = F_{j+1}$. The series will obviously converge satisfactorily with the value of j given in (26) since $\|A_c T/r\| < 1$. Once F has been calculated, E can be obtained from (16) and the ZOH model from (11).

This development leads to the following computational procedure:

1. Define the matrix A_c, scalar T and integers j and N. The integer j should satisfy (26) and a suggested value for N is 16.
2. Calculate F_1 and T_1 using (25).
3. For $k = 1, 2, \ldots, j$ calculate F_{k+1} recursively using (24).
4. Set $F_{j+1} \Rightarrow F$, $I + FA_cT \Rightarrow E$ and $I + EA_cT \Rightarrow A_d$.

Algorithm (F-E-A_d) results in a non-standard five matrix model, but it is easily converted to a standard model as given in (4), [8].

Continualization Procedures

The reverse process of converting from a discrete-time model to an *equivalent* continuous-time model will now be considered, *i.e.* converting between the model in (4) and the model in (2), $R_d \rightarrow R_c$ in the *ZOH* case, or between (16) and (2), $R_{df} \rightarrow R_c$ in the *FOH* sense. Of course, by itself R_d has no information regarding the signal values between samples and model conversion in this direction must be taken in context.

The algorithms for continualization require *logarithmic* operations instead of matrix exponentiation. When $(A_d - I)$ or A_c is invertible, it is easily concluded that the matrices of R_c in (2) are inverted from (5) and (7) by:

$$A_c = \frac{1}{T}\ln(A_d)\,, \quad B_c = (A_d - I)^{-1}A_c B_d \tag{27}$$

with the understanding that $C_c = C_d$ and $D_c = D_d$ as before, thus completing the continuous-time model in (2). The method is given in the following. The Taylor series expansion for the function $ln(x)$ in the neighborhood of $x = 1$ leads to

$$A_c = -\frac{1}{T}\sum_{i=1}^{\infty}\frac{(I - A_d)^i}{i} \tag{28}$$

The problem of using a truncated version of (28), [9], is that for matrices A_d with

$$|\lambda_{\max}| > 0.5 \tag{29}$$

where $\lambda_{\max}$ is the maximum magnitude eigenvalue of $(A_d - I)$, the series may require large

N, leading to considerable round-off errors if it converges at all. This algorithm resolves this problem by using the following basic property of the logarithm function.

$$\ln(x) = r\ \ln\left[(x)^{1/r}\right] = r\sum_{i=1}^{\infty}\frac{(x^{1/r}-1)^i}{i}(-1)^{(i+1)} = -r\sum_{i=1}^{\infty}\frac{(1-x^{1/r})^i}{i} \tag{30}$$

With this approach the truncated series for calculation becomes

$$A_c = -\frac{r}{T}\sum_{i=1}^{N}\frac{(I-A_d^{1/r})^i}{i} \tag{31}$$

where the integer j satisfies that

$$|\,\lambda(A_d^{1/r}-I)\,|_{\max} < 0.5\ , \quad with\ \ r = 2^j \tag{32}$$

A computational algorithm, referred to as Algorithm *Ln*, calculating A_c according to (31) is given below. It has been experimentally verified that the accuracy of using (31) is satisfactory even for matrices A_d where some eigenvalues of $L = A_d - I$ have magnitude greater than one.

Having determined A_c, the remaining matrices in the C-T ZOH equivalent state space model (2) could be calculated using (27) if A_c is nonsingular. If, however, A_c is singular, then the matrices E and F, appearing in (11) and (14) should be calculated first. It follows that $C_c = C_d$, $D_c = D_d$ and

$$B_c = \frac{1}{T}E^{-1}B_d \tag{33}$$

for the continualization of the ZOH model.

It is easily determined that the C-T model in (2) can be obtained from the five matrix model in (9) by using the Algorithm *Ln* to calculate A_c and from the availability of F in Algorithm *F-E-A_d*, inverting (14) to get

$$B_c = \frac{1}{T}F^{-1}B_{d1} = \frac{1}{T}(E-F)^{-1}B_{d0} \tag{34}$$

The required five matrix D-T model (9) could be obtained from a standard four matrix D-T model in a straightforward manner, see [8] for details.

Algorithm: (*Ln*) Continualization of Models In order to calculate A_c defined by (28), it is first necessary to determine integers j and r satisfying (32). This is done as specified by the first five steps of the procedure below. Using (31),

1. Define the matrix A_d, scalar T and integer N. A suggested value for N is *25*.
2. Set $0 \Rightarrow j$ and $A_d \Rightarrow A_j$.
3. Set $I - A_j \Rightarrow L$.
4. If $|\lambda(L)|_{max} < 0.5$, go to 6, else go to 5.
5. Set $j+1 \Rightarrow j\ ;\ \ (A_j)^{1/2} \Rightarrow A_j$ and go to 3.

6. Set $2^j \Rightarrow r$ and calculate A_c using $A_j = A_d^{1/r}$ in (31), i.e.

$$A_c = -\frac{r}{T}\sum_{i=1}^{N}\frac{(I-A_j)^i}{i}$$

Conclusions

Newly developed numerically robust algorithms have been presented. These algorithms deals with the often encountered problem of *discretization* of continuous-time models as well as the inverse problem of recreating a continuous-time model from a given discrete-time model. In addition to the standard *ZOH* procedure, a method called the *FOH* method that is equivalent to the next higher order approximation up from the *ZOH*, representing a piecewise linear approximation to the input functions. With these algorithms the design engineer has an increased flexibility to move between the continuous and discrete model domains.

References

[1] VanLandingham, H. F. (1985), *Introduction to Digital Control Systems*, Macmillan Pub. Co., New York, NY.

[2] Brogan, W. L. (1984), *Modern Control Theory*, Prentice-Hall Pub. Co., Englewood Cliffs, NJ.

[3] Bingulac, S. and N. K. Sinha (1990), "On the Identification of Continuous-Time Multivariable Systems", *Math. Comput. Modeling*, Vol. 14, pp, 203-208.

[4] Strmcnik, S. and F. Bremsak (1979), "Some New Transformation Algorithms on the Identification of Continuous-Time Multivariable Systems Using Discrete Identification Methods", *Proc. 5th IFAC Symposium on Identification and System Parameter Estimation*, Darmstad, Germany, pp. 397-405.

[5] Laub, A. J. (1985), "Numerical Linear Algebra Aspects of Control Design Computations", *IEEE Trans. on Automatic Control*, Vol. AC-30, No. 2, pp. 97-108.

[6] Sinha, N. K. and B. Kusta (1983), *Modeling and Identification of Dynamic Systems*, Van Nostrand-Reinhold Publishers, New York, NY.

[7] Bingulac, S. and D. Cooper (1991), "Use of Pseudo-Observability Indices in Identification of Continuous-Time Multivariable Models", *Identification of Continuous-Time Systems*, Sinha, N. K. and G. P. Rao, (editors), Kluver Academic Publishers, The Netherlands.

[8] Bingulac, S. and H.F. VanLandingham (1991), "A Unified Approach to the Discretization and 'Continualization' of MIMO Systems", Unpublished Research Report, EE Dept., Virginia Tech, Blacksburg, VA.

[9] Lastman, G.J., S.C. Puthenpura and N.K. Sinha (1984), "Algorithm for the Identification of Continuous-Time Multivariable Systems from Their Discrete-Time Models", *Electronics Letters*, Vol. 20, No. 22, pp. 918-919.

Boundary Value Problem of Linear Weakly Coupled Systems

Muhammad T. Qureshi and *Zoran Gajic*
Rutgers University, Department of Electrical and
Computer Engineering, Piscataway, NJ 08855-0909

Abstract

This paper treats the general boundary value problem of the weakly coupled linear dynamical systems. The existence of the solution in terms of the reduced-order completely decoupled dynamical systems is established.

1 Introduction

Consider the general boundary value problem for a linear weakly coupled system consisting of $n_1 + n_2$ differential equations

$$\dot{x} = A_1(t)x + \epsilon A_2(t)z \tag{1}$$

$$\dot{z} = \epsilon A_3(t)x + A_4(t)z \tag{2}$$

and $n_1 + n_2$ boundary conditions

$$M(\epsilon)\begin{bmatrix} x(0,\epsilon) \\ z(0,\epsilon) \end{bmatrix} + N(\epsilon)\begin{bmatrix} x(1,\epsilon) \\ z(1,\epsilon) \end{bmatrix} = \begin{bmatrix} c_1(\epsilon) \\ c_2(\epsilon) \end{bmatrix} \tag{3}$$

on the interval $0 \leq t \leq 1$. Here $x \in R^{n_1}$, $z \in R^{n_2}$, $c_1 \in R^{n_1}$, $c_2 \in R^{n_2}$, and $A_1(t), A_2(t), A_3(t), A_4(t), M(\epsilon), N(\epsilon)$ are matrices of appropriate dimensions. A small parameter ϵ couples the states x and z. It is assumed that $A_1(t), A_2(t), A_3(t)$ and $A_4(t)$ are continuous function for $0 \leq t \leq 1$, and $M(\epsilon) = M(0) + O(\epsilon)$, $N(\epsilon) = N(0) + O(\epsilon)$, $c_i(\epsilon) = c_i(0) + O(\epsilon), i = 1, 2$.

In [1], Chang solved a similar boundary value problem for singularly perturbed systems, where a transformation approach is used to block diagonalize the system. In this paper we will use the transformation proposed in [2]-[3] to block diagonalize (1)–(2). The application of that transformation will simplify the analytical and computational treatment of (1)–(2) in terms of the reduced-order completely decoupled dynamical systems. The presented results are important for the optimal filtering and control problems where the boundary value problems defined by (1)–(3) appear. The weakly coupled systems have been studied in the control theory by many researchers [3]–[15]. Its practical importance has been demonstrated in the recent monograph [4], where many real physical control problems of weakly coupled systems were considered.

2 Main Results

Applying the transformation defined in [2]–[3] as

$$\begin{bmatrix} \eta \\ \zeta \end{bmatrix} = \begin{bmatrix} I & -\epsilon L(t) \\ -\epsilon H(t) & I \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} = T(t,\epsilon) \begin{bmatrix} x \\ z \end{bmatrix} \tag{4}$$

to (1)-(2) we obtain an equivalent block diagonalized form

$$\dot{\eta} = A_{10}(t)\eta \tag{5}$$

$$\dot{\zeta} = A_{40}(t)\zeta \tag{6}$$

where

$$A_{10}(t) = A_1(t) - \epsilon^2 L(t) A_3(t) \tag{7}$$

$$A_{40}(t) = A_4(t) - \epsilon^2 H(t) A_2(t), \tag{8}$$

and the transformation matrices satisfy the following Riccati type differential equations

$$\dot{L}(t) = A_1(t)L(t) - L(t)A_4(t) + A_2(t) - \epsilon^2 L(t)A_3(t)L(t) \tag{9}$$

$$\dot{H}(t) = A_4(t)H(t) - H(t)A_1(t) + A_3(t) - \epsilon^2 H(t)A_2(t)H(t). \tag{10}$$

The existence of a nonsingular T in (4) is ensured for sufficiently small ϵ which makes it diagonally dominant. Note that no initial or terminal conditions are imposed on (9)–(10).

Since the differential Riccati equation, in general, has a finite escape time [16], we have to establish the existence of bounded solutions for (9) and (10). The only assumption we are imposing is that $A_1(t), A_2(t), A_3(t)$, and $A_4(t)$ are continuous function on $0 \leq t \leq 1$. Due to the same structure of (9) and (10) it suffices to show that either one of them possesses a bounded solution. The following lemma is proved for the existence of such a solution.

LEMMA: There exists $\epsilon_0 > 0$ such that equations (9) and (10) have solutions $L(t)$, and $H(t)$, respectively, which are uniformly bounded for $0 \leq t \leq 1$ in the region $0 < \epsilon \leq \epsilon_0$.

PROOF: Let $X(t)$ be the fundamental matrix of $\dot{x} = A_1(t)x$, and $Z(t)$ be the fundamental matrix of $\dot{z} = A_4(t)z$. Since $A_1(t)$ and $A_4(t)$ are continuous on [0,1], and therefore bounded on [0,1], there exist $\sigma_1 > 0$ and $\sigma_2 > 0$ such that

$$||A_1(t)|| < \sigma_1$$

$$||A_4(t)|| < \sigma_2,$$

which implies [1]

$$\begin{aligned} ||X(t)X^{-1}(s)|| &\leq K_1 exp(\sigma_1|t-s|) \quad 0 \leq t, s \leq 1 \\ ||Z(t)Z^{-1}(s)|| &\leq K_2 exp(\sigma_2|t-s|) \quad 0 \leq t, s \leq 1 \end{aligned} \tag{11}$$

where K_1, K_2 are positive constants.

It can be verified by differentiation that

$$L(t) = \int_0^t X(t)X^{-1}(s)[A_2(s) - \epsilon^2 L(s)A_3(s)L(s)]Z(s)Z^{-1}(t)ds \tag{12}$$

is a solution of (9). To show that this is a bounded solution, let $||L(t)|| \leq \rho$, with ρ being chosen later, then

$$||L(t)|| \leq \int_0^t K_1K_2e^{\sigma_1|t-s|}e^{\sigma_2|t-s|}ds.(||A_2(t)|| + \epsilon^2\rho^2||A_3(t)||).$$

Since in the integration we can take $s \leq t$, or $|t-s| = t-s$, we obtain by evaluating the integral

$$||L(t)|| \leq \frac{K_1K_2}{\sigma_1+\sigma_2}\left[e^{(\sigma_1+\sigma_2)t} - 1\right](||A_2(t)|| + \epsilon^2\rho^2||A_3(t)||).$$

The inequality is still maintained if we take $t = 1$ in the square brackets, giving

$$||L(t)|| \leq \alpha(||A_2(t)|| + \epsilon^2\rho^2||A_3(t)||),$$

where

$$\alpha = \frac{K_1K_2}{\sigma_1+\sigma_2}\left[e^{(\sigma_1+\sigma_2)} - 1\right].$$

Now we have to pick ρ and ϵ_0 such that $||L(t)|| \leq \rho$. Pick

$$\rho = 2\alpha||A_2(t)||,$$

so

$$||L(t)|| \leq \frac{\rho}{2} + \alpha\epsilon^2\rho^2||A_3(t)||.$$

Pick ϵ_0 such that

$$\alpha\epsilon_0^2\rho^2||A_3(t)|| \leq \frac{\rho}{2}$$

or

$$\epsilon_0^2 \leq \frac{1}{4\alpha^2||A_2(t)||||A_3(t)||}.$$

Thus

$$||L(t)|| \leq \frac{\rho}{2} + \frac{\rho}{2} < \rho.$$

Therefore $L(t)$ has a bounded solution in $||L(t)|| \leq \rho$. Since [0,1] is a compact interval, the solution is uniformly bounded. On the same lines it can be proved that $H(t)$ also has a uniformly bounded solution.

Consequently, the change of variables (4) transforms the system (1)–(2) to (5)–(6). Applying (4) to the boundary conditions (3), we obtain

$$\hat{M}(\epsilon)\begin{bmatrix} \eta(0,\epsilon) \\ \zeta(0,\epsilon) \end{bmatrix} + \hat{N}(\epsilon)\begin{bmatrix} \eta(1,\epsilon) \\ \zeta(1,\epsilon) \end{bmatrix} = \begin{bmatrix} c_1(\epsilon) \\ c_2(\epsilon) \end{bmatrix} \tag{13}$$

where $\hat{M}(\epsilon) = M(\epsilon)T^{-1}(0,\epsilon)$, $\hat{N}(\epsilon) = N(\epsilon)T^{-1}(1,\epsilon)$, and $T^{-1}(t,\epsilon)$ is given by

$$T^{-1}(t,\epsilon) = \begin{bmatrix} I + \epsilon^2 L(t)(I - \epsilon^2 H(t)L(t))^{-1}H(t) & \epsilon L(t)(I - \epsilon^2 H(t)L(t))^{-1} \\ \epsilon(I - \epsilon^2 H(t)L(t))^{-1}H(t) & (I - \epsilon^2 H(t)L(t))^{-1} \end{bmatrix}.$$

Therefore the equivalent transformed boundary value problem is described by (5), (6), and (13).

Let $\mathcal{E}(t,\epsilon)$ be the fundamental matrix of (5), and $\mathcal{Z}(t,\epsilon)$ be the fundamental matrix of (6), then we have the following solution for (5)–(6)

$$\begin{bmatrix} \eta(t) \\ \zeta(t) \end{bmatrix} = \begin{bmatrix} \mathcal{E}(t,\epsilon) & 0 \\ 0 & \mathcal{Z}(t,\epsilon) \end{bmatrix}\begin{bmatrix} \alpha_1(\epsilon) \\ \alpha_2(\epsilon) \end{bmatrix} \tag{14}$$

where $\alpha_1(\epsilon)$, and $\alpha_2(\epsilon)$ are arbitrary constants. It only remains to choose $\alpha_1(\epsilon)$, and $\alpha_2(\epsilon)$ to satisfy the boundary conditions (13). Substituting (14) into (13) yields

$$\Delta(\epsilon)\begin{bmatrix}\alpha_1(\epsilon)\\ \alpha_2(\epsilon)\end{bmatrix} = \begin{bmatrix}c_1(\epsilon)\\ c_2(\epsilon)\end{bmatrix}$$

where

$$\Delta(\epsilon) = \hat{M}(\epsilon)\begin{bmatrix}\mathcal{E}(0,\epsilon) & 0\\ 0 & \mathcal{Z}(0,\epsilon)\end{bmatrix} + \hat{N}(\epsilon)\begin{bmatrix}\mathcal{E}(1,\epsilon) & 0\\ 0 & \mathcal{Z}(1,\epsilon)\end{bmatrix}. \tag{15}$$

If $\Delta^{-1}(\epsilon)$ exists then

$$\begin{bmatrix}\alpha_1(\epsilon)\\ \alpha_2(\epsilon)\end{bmatrix} = \Delta^{-1}(\epsilon)\begin{bmatrix}c_1(\epsilon)\\ c_2(\epsilon)\end{bmatrix}$$

and thus (14) is a solution to the boundary value problem.

Note that as $\epsilon \to 0$, $T^{-1}(t,\epsilon) \to I$, and therefore $\lim_{\epsilon\to 0}\hat{M}(\epsilon) = M(0)$, and $\lim_{\epsilon\to 0}\hat{N}(\epsilon) = N(0)$. Therefore,

$$\lim_{\epsilon\to 0}\Delta(\epsilon) = M(0)\begin{bmatrix}\mathcal{E}(0,0) & 0\\ 0 & \mathcal{Z}(0,0)\end{bmatrix} + N(0)\begin{bmatrix}\mathcal{E}(1,0) & 0\\ 0 & \mathcal{Z}(1,0)\end{bmatrix}$$

where $\mathcal{E}(t,0) = \lim_{\epsilon\to 0}\mathcal{E}(t,\epsilon)$, and $\mathcal{Z}(t,0) = \lim_{\epsilon\to 0}\mathcal{Z}(t,\epsilon)$. Hence for sufficiently small ϵ, the inverse $\Delta^{-1}(\epsilon)$ exists if we make the following assumption: The matrix

$$\lim_{\epsilon\to 0}\Delta(\epsilon) \equiv \Delta(0) = M(0)\begin{bmatrix}\mathcal{E}(0,0) & 0\\ 0 & \mathcal{Z}(0,0)\end{bmatrix} + N(0)\begin{bmatrix}\mathcal{E}(1,0) & 0\\ 0 & \mathcal{Z}(1,0)\end{bmatrix} \tag{16}$$

is nonsingular. It is very well known fact that for the optimal linear control problem, the matrices $M(0)$ and $N(0)$ are given by

$$M(0) = \begin{bmatrix}I & 0\\ 0 & 0\end{bmatrix}, \quad and \quad N(0) = \begin{bmatrix}0 & 0\\ 0 & I\end{bmatrix}$$

so that

$$\Delta(0) = \begin{bmatrix}\mathcal{E}(0,0) & 0\\ 0 & \mathcal{Z}(1,0)\end{bmatrix}.$$

But the fact that the system matrices $\mathcal{E}(0,0)$ and $\mathcal{Z}(1,0)$ are nonsingular, so does $\Delta(0)$.

Consequently, the following theorem summarizes the results.

THEOREM: Let $A_1(t), A_2(t), A_3(t)$, and $A_4(t)$ be continuous functions on $0 \leq t \leq 1$, and let assumption (16) holds, then for all sufficiently small ϵ the boundary value problem (5), (6), and (13) has the solution

$$\begin{bmatrix} \eta(t,\epsilon) \\ \zeta(t,\epsilon) \end{bmatrix} = \begin{bmatrix} \mathcal{E}(t,\epsilon) & 0 \\ 0 & \mathcal{Z}(t,\epsilon) \end{bmatrix} \Delta^{-1}(\epsilon) \begin{bmatrix} c_1(\epsilon) \\ c_2(\epsilon) \end{bmatrix}$$

for $0 \leq t \leq 1$, where $\Delta(\epsilon)$ is given by (15). In the original variables, for all sufficiently small ϵ the boundary value problem (1)–(3) has the solution

$$\begin{bmatrix} x(t,\epsilon) \\ z(t,\epsilon) \end{bmatrix} = \begin{bmatrix} I + \epsilon^2 L(t)P(t,\epsilon)H(t) & \epsilon L(t)P(t,\epsilon) \\ \epsilon P(t,\epsilon)H(t) & P(t,\epsilon) \end{bmatrix} \begin{bmatrix} \eta(t,\epsilon) \\ \zeta(t,\epsilon) \end{bmatrix}$$

where $P(t,\epsilon) = (I - \epsilon^2 LH)^{-1}$ and L(t) and H(t) are the solutions of (8) and (9), respectively.

3 Conclusion

The general boundary value problem is solved for weakly coupled linear dynamical systems. The transformation approach is used to establish the existence of the solution. A new version of the Chang's transformation [1] is used, which leads to simpler analysis and the ability to obtain completey decoupled equations.

References

[1] K. W. Chang, "Singular perturbations of a general boundary value problem", *SIAM J. Math. Anal.,* vol. 3, pp. 520–526, Aug. 1972.

[2] M. Qureshi and Z. Gajic, "A new version of the Chang transformation", *IEEE Tran. Aut. Control,* vol. AC-36, Nov. 1991.

[3] M. Qureshi, *Parallel Algorithms for Discrete Singularly Perturbed and Weakly Coupled Filtering and Control Problems*, Ph.D Thesis, Rutgers University, New Jersey, 1991.

[4] Z. Gajic, D. Petkovski and X. Shen, *Singularly Perturbed and Weakly Coupled Linear Control Systems – A Recursive Approach.* Springer Verlag, Berlin–New York, 1990.

[5] J. D. Delacour, M. Darwish and J. Fantin, "Control strategies of large-scale power systems", *Int. J. Control*, vol 27, pp. 753–767, 1978.

[6] N. Harkara, Dj. Petkovski and Z. Gajic, "The recursive algorithm for optimal output feedback control problem of linear weakly coupled systems", *Int. J. Control*, vol. 50, pp. 1–11, 1989.

[7] T. Ishimatsu, A. Mohri and M. Takata, "Optimization of weakly coupled systems by a two-level method", *Int. J. Control*, vol. 22, pp. 877–882, 1975.

[8] H. Khalil, and P. Kokotovic, "Control strategies for decision makers using different models of the same system", *IEEE Tran. Aut. Control*, vol. AC-23, pp. 289–298, 1978.

[9] P. Kokotovic, W. Perkins, J. Cruz Jr., and D'Ans, "ϵ-coupling for near optimum design of large scale linear systems", *Proc. IEE*, vol. 116, pp. 889–992, 1969.

[10] M Mahmoud, "A quantitative comparison between two decentralized control approaches", *Int. J. Control*, vol. 28, pp. 261–275, 1978.

[11] B. Petrovic and Z. Gajic, "Recursive solution of linear quadratic nash games for weakly interconnected systems", *J. Optimization Theory and Applications*, vol. 56, pp. 463–477, 1988.

[12] M. Sezer, and D. Siljak, "Nested ϵ-decomposition and clustering of complex systems", *Automatica*, vol. 22, pp. 321-331, 1986.

[13] H. D. Washburn, and J. Mendel, "Mutistage estimation of dynamical and weakly coupled systems in continuous time linear systems", *IEEE Trans. Aut. Control*, vol. AC-25, pp. 71–76, 1980.

[14] X. Shen and Z. Gajic, "Near optimum steady state regulators for stochastic linear weakly coupled systems", *Automatica*, vol. 26, pp. 919–923, 1990.

[15] X. Shen and Z. Gajic, "Optimum reduced-order solution of the weakly coupled discrete Riccati equation", *IEEE Tran. Aut. Control*, vol. AC-35, pp. 600–602, 1990.

[16] T. Sasagawa, "On the finite escape phenomena for matrix Riccati equation", *IEEE Tran. Aut. Control*, vol. AC-27, pp. 977–979, 1982.

Single Loop Stability Margins for Multirate Control Systems

David S. Flamm
Department of Electrical Engineering, Princeton University
Princeton, NJ 08544

Abstract

After reviewing the difficulties in computing stability margins for multirate digital systems, we make appropriate definitions and show how to compute the stability radius for periodic perturbations of such systems. This computation requires the use of a non-standard time-invariant representation of periodic linear systems, since the conventional representation and computations lead to the possibility of non-causal destabilizing perturbations and hence conservative values. We also note that gain and phase margin computation can be pefomed using the the standard time-invariant model, by means of the multivariable Nyquist locus.

1 Introduction

The validation of a control system design requires the computation of stability margins with respect to errors in the model of the system to be controlled. In the absence of a practical technique for computing stability margins with respect to simultaneous variations in real plant parameters, it is common for control practitioners to compute single loop gain and phase margins at points in a feedback system at which model variations have physical significance. Typically it is desirable to do these computations in the frequency domain, so that well-developed engineering interpretations can be applied.

Many practical control designs are implemented as multirate systems in order to utilize hardware as efficiently as possible when computing power presents a limitation on design. Some techniques for the frequency domain analysis of multirate systems are limited by the fact that the natural models for multirate systems are not time-invariant but periodically time-varying. It is only when a loop is broken at a point at which the sampling rate is the greatest common divisor of all the sampling rates in the system that the natural input-output model is time-invariant.

These limitations were partially addressed in the 1950's by the work of Kranc [6] and Kalman-Bertram [4]. This work may be summarized by saying that by breaking the loop at any point and accumulating all inputs and ouputs over a time interval for which the overall system is periodic into a vector input and a vector output one can obtain a time-invariant model. However, this time-invariant system is not readily amenable to stability margin calculation.

In certain special cases, one version of margins can be calculated by shifting gain and phase perturbations to a slow-rate point using block diagram manipulations. Then the problem is reduced to the analysis of the transfer function of a single-input/single-output system. However in the general case, this cannot be done, and the time invariant system to be analyzed is inherently a multi-input/multi-output (MIMO) system, *even though the physical "broken loop" under consideration has a single input and a single output* (a SISO system).

The obvious approach, to the control analyst of "modern" inclinations, is to compute the "principal gain" of this MIMO system, recognizing this as a lower bound for the gain margin of the time-varying loop. Unfortunately, the natural destabilizing perturbation corresponding to this margin will generally be a non-causal perturbation of the original system.

In this paper we show that the calculation just mentioned gives a stability margin which is non-conservative is the sense that there is a periodic perturbation within the computed stability radius which destabilizes the closed loop. (This perturbation *will* be conservative in the sense that a model of the perturbing system may have complex valued parameters.)

2 Problem Formulation

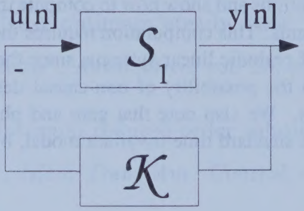


Figure 1: Periodic Feedback System

We assume that we have a feedback system as in Figure 1. Here S_1 is a periodic time varying linear system in discrete time, given by the equations

$$x[n+1] = A_n x[n] + B_n u[n] \tag{1}$$

$$y[n] = C_n x[n] + D_n u[n] \tag{2}$$

where $x[i] \in \mathbb{R}^p$, $u[i] \in \mathbb{R}^q$ and $y[i] \in \mathbb{R}^r$, for $i = 0, 1, \ldots$. We assume that this system has period N, so that $A_i = A_{i+N}$, $B_i = B_{i+N}$, etc. The system $\mathcal{K}$ is assumed to be a linear system which may have additional properties as discussed below. $\mathcal{K}$ is a perturbation of the identity system, and we use it to define and analyze stability margins.

Such models cover a large class of "hybrid" (interconnected continuous- and discrete-time) systems. Using the methods presented by Kalman-Bertram [4] one can write down time invariant models for these systems. Such models have been recently presented in a more elegant mathematical framework in the papers [5] and [3]. This class includes multirate sampled-data systems, as considered, for example, by [1].

For the present purposes , we assume that the feedback loop represented in Figure 1 is a scalar signal which we have "pulled out" of the overall system in order to do a stability margin analysis. In other words, we assume $q = r = 1$. We shall call this the *periodic single-input/single-output* (PSISO) case.

The problem we consider is the following: Assuming the closed loop system is stable if $\mathcal{K} \equiv 1$, find the largest $\mu > 0$ (the *margin*) such that the closed loop system is stable for all stable $\mathcal{K}$ (subject to additional assumptions below) satisfying $\|1 - \mathcal{K}\| < \mu$. Here $\|\cdot\|$ is the l^2-induced operator norm.

2.1 Classes of Perturbations

There are three sets of assumptions about $\mathcal{K}$ which are of interest:

1. $\mathcal{K}$ is memoryless. It therefore has a transfer function K which may be a real constant, a unit-magnitude complex number, or a general complex constant. These possibilities result in what may be called the *gain margin, phase margin* and *m-circle radius* problems, respectively. However the m-circle problem does not have the same relationship with time-invariant perturbations as does the m-circle problem for time-invariant systems, as we discuss below.

2. $\mathcal{K}$ is a time-invariant system with transfer function $K(z)$, which we assume to be stable. This is the *time-invariant perturbation* stability problem. Here we assume that the sampling interval associated with $K(z)$ is the sample period associated with the discrete time signal at the loop-breaking point.

3. $\mathcal{K}$ is a periodic system with period N. We call this the *periodic perturbation* stability margin problem.

Note that we have listed these problems is order of decreasing conservativeness. That is, the periodic perturbation stability margin is bounded above by the time-invariant perturbation stability margin, which is in turn bounded above by the gain margin.

Assumption (1.) uses the common terminology "m-circle", but it is not clear that this is the correct generalization to periodic systems of the time-invariant notion of m-circle. In the case of time-invariant systems, for every time-invariant destabilizing perturbation of the feedback loop there is a constant perturbation of the same norm which is also destabilizing. Thus to find the norm of the smallest destabilizing perturbation (the m-circle problem), it suffices to only consider constant perturbations. In the case of periodic systems, this property does not hold, so we define the time-invariant perturbation stability margin problem (2.), which is a more meaningful generalization of the m-circle test. Problem (1.) of dubious interest for periodic systems, since it is not clear what relationship these quantities may have with properties of actual perturbations, unless the perturbations really are constant. Problem (1.) is solved in [8]. We do not have a solution for problem (2.), but we do for problem (3.).

3 Difficulties in Stability Margin Computation

In order to apply frequency domain methods of stability margin computation, one needs a time-invariant system. The standard time-invariant representation of equations (1) and (2) is obtained by "stacking" the inputs and outputs of $\mathcal{S}_1$ over periods of N samples, as indicated in equations (3-4).

$$\begin{aligned} \tilde{x}[n+1] &= \tilde{A}\tilde{x}[n] + \tilde{B}\tilde{u}[n] \qquad (3) \\ \tilde{y}[n] &= \tilde{C}\tilde{x}[n] + \tilde{D}\tilde{u}[n] \qquad (4) \end{aligned}$$

with

$$\tilde{y}[n] \triangleq \begin{pmatrix} y[nN] \\ y[nN+1] \\ \vdots \\ y[nN+N-1] \end{pmatrix}$$

$$\tilde{u}[n] \triangleq \begin{pmatrix} u[nN] \\ u[nN+1] \\ \vdots \\ u[nN+N-1] \end{pmatrix}$$

$$\tilde{x}[n] \triangleq x[nN]$$

$\tilde{A}$, $\tilde{B}$, etc. follow from these definitions, and are given, for example, in [5]. This "stacked" model constitutes a MIMO time-invariant system although the original system (in this paper) is a SISO periodic system.

There are two types of stability analyses commonly computed for MIMO systems when they are time-invariant:

1. Most commonly, control practitioners select a scalar loop within the MIMO system which is expected to suffer model errors or variations in behavior during operation. For this single variable loop, gain and phase margin computations are then performed using frequency domain techniques.

   In the case of multirate control systems, the SISO model obtained by "breaking" the loop of interest will not generally be time-invariant. It is only when the sampling rate at this point is the greatest common divisor of all sampling rates that time-invariance will hold.

   If the signals at this loop-breaking point do not have this "slow" sample rate, but there *is* a slow rate point after some cascade of systems in the same loop, it is still possible to compute gain and phase margins as defined in problem (1) above by shifting the constant multiplier defining the margin to the slow rate point, and doing the computations for the corresponding time-invariant system.

   In the case of general multirate systems such shifting cannot be done.

2. It is also increasingly common to define the multivariable gain margin for a time-invariant MIMO system at a multivariable point in the system as the smallest norm of all destabilizing multiplicative perturbations at that point. This computation is performed by finding the largest frequency-dependent value of the largest singular value of the complementary sensitivity matrix tranfer function. The reciprocal of this quantity is the natural multivariable extension of the m-circle test for SISO systems.

   Performing this computation for the stacked model of a periodic system will result in a destabilizing constant perturbation of the transfer function matrix. In the context of the stacked repesentations of periodic linear systems constant transfer function matrices will generally be non-causal. Causal constant transfer function matrices for such systems are lower triangular.

4 Solutions

4.1 Gain Margin

Problem (1) is readily solved by using multivariable Nyquist notions, as apparently first suggested for gain and phase margins in [8]. The idea is that for the case of a single loop of a periodic system we consider here, we are only considering perturbing a single physical gain. The transformation to a multivariable problem is a modeling construct, and the original single gain change becomes transformed into an identical gain change in all the (fictitious) loops. Let

$S_1(z)$ denote the transfer function matrix associated with the "stacked" representation of S_1. The m-circle radius may be computed by finding the eigenvalue of $R_1(z) \stackrel{\Delta}{=} S_1(z)(I_{N\times N} + S_1(z))^{-1}$ of smallest magnitude, for $z = e^{j\omega}$, $\omega \in [0, 2\pi]$. Phase margin and gain margin are computed similarly, as detailed in [8].

For SISO time-invariant systems, the m-circle radius is also the norm of the smallest destabilizing time-invariant perturbation. The same holds for the singular value generalization to MIMO time-invariant systems. However the above generalization to PSISO systems does *not* give the norm of either the smallest destabilizing time-invariant perturbation or the smallest destabilizing periodic perturbation. This is because constant-gain perturbations in a scalar loop are multiples of the identity matrix in the stacked form of the PSISO problem, whereas time-invariant or periodic systems have a more general form.

4.2 Periodic Perturbation Stability Margin

In this section, we shall show that the stability radius computation for MIMO time-invariant systems applied to the stacked form of a PSISO system also gives the stability radius for periodic perturbations, in spite of the suggestion of non-causality at the end of Section 3.

Let $S_1(s)$ denote the transfer function matrix of the stacked form of the PSISO system $\mathcal{S}_\infty$. Treating this as an ordinary MIMO system, it is well-known that the smallest norm of the all destabilizing stable matrix transfer functions is the reciprocal of the largest frequency-dependent singular value $\overline{\sigma}(S_1(z)(I_{N\times N} + S_1(z))^{-1})$ of the complementary sensitivity matrix, $R_1(z)$. A corresponding destabilizing perturbation K_1 can be constructed by taking the outer product of the corresponding left and right singular vectors, just as in Section 4.3 below. The resulting constant matrix will be lower triangular (and hence causal) only in very special cases (if the first non-zero entry in the left singular vector is the m^{th}, then the $(m+1)^{st}$ and succeeding entries in the right singular vector must be zero).

However, it is easy to construct a non-constant transfer function matrix having the same norm which is the stacked version of a causal PSISO system. The necessary and sufficient condition for $P(z)$ to be the stacked form of a PSISO system is that $P(\infty)$ be lower triangular. Let $\overline{\omega}$ be the frequency at which the complementary sensitivity attains its norm. If we define the scalars k_{ij} by $K_1 = [k_{ij}]$, then $K_2(z) \stackrel{\Delta}{=} [k_{ij} e^{j\overline{\omega}} z^{-1}]$ is the transfer function of the stacked representation of a causal PSISO system which satisfies $\|K_2\|_\infty = \|K_1\|_\infty$ and $K_2(e^{j\overline{\omega}}) = K_1$. Except for this construction of K_2, the proof that these calculations are just as in Section 4.3.

4.3 A Simpler Computation

In order to compute the periodic perturbation stability margin using the stacked model, one must first compute the matrices $\tilde{A}$, $\tilde{B}$, etc., which in general can be burdensome.

In this section, we show that the time-invariant representation of periodic systems presented in [2] can also be used to find the periodic perturbation stability margin, with simpler calculations.

This model is defined as follows:

$$\left.\begin{array}{rcl} \overline{x}[i+1] & = & \overline{A}\overline{x}[i] + \overline{B}\overline{u}[i] \\ \overline{y}[i] & = & \overline{C}\overline{x}[i] + \overline{D}\overline{u}[i] \end{array}\right\} \qquad (5)$$

where

$$\overline{A} = \begin{bmatrix} 0 & 0 & \cdots & 0 & A_N \\ A_1 & 0 & \cdots & 0 & 0 \\ 0 & A_2 & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & 0 \\ 0 & \cdots & 0 & A_{N-1} & 0 \end{bmatrix}, \quad \overline{B} = \begin{bmatrix} 0 & 0 & \cdots & 0 & B_N \\ B_1 & 0 & 0 & \cdots & 0 \\ 0 & B_2 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & 0 & 0 \\ 0 & 0 & \cdots & B_{N-1} & 0 \end{bmatrix},$$

$$\overline{C} = \begin{bmatrix} C_1 & 0 & \cdots & 0 & 0 \\ 0 & C_2 & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & C_{N-1} & 0 \\ 0 & 0 & \cdots & 0 & C_N \end{bmatrix}, \quad \overline{D} = \begin{bmatrix} D_1 & 0 & \cdots & 0 & 0 \\ 0 & D_2 & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & D_{N-1} & 0 \\ 0 & 0 & \cdots & 0 & D_N \end{bmatrix},$$

$$\overline{x} = \begin{bmatrix} \overline{x}_1 \\ \overline{x}_2 \\ \vdots \\ \overline{x}_N \end{bmatrix}, \quad \overline{u} = \begin{bmatrix} \overline{u}_1 \\ \overline{u}_2 \\ \vdots \\ \overline{u}_N \end{bmatrix} \quad \text{and} \quad \overline{y} = \begin{bmatrix} \overline{y}_1 \\ \overline{y}_2 \\ \vdots \\ \overline{y}_N \end{bmatrix};$$

$\overline{x}_i$, $\overline{u}_i$, and $\overline{y}_i$ have dimensions p, q and r, respectively, and $\overline{x}$, $\overline{u}$ and $\overline{y}$ have dimensions $p \cdot N$, $q \cdot N$ and $r \cdot N$, respectively. It is easy to see that this time-invariant system amounts to N copies of the original system running in parallel, offset in time successively by one time step. For this reason, we shall call this representation of a periodic system the "staggered" representation.

One way to embed the periodic system defined in equations (1) and (2) in this model is the following: For each integer $k \geq 0$ let the integer i_k be that one which satisfies $1 \leq i_k \leq N$ and $i_k = (k-1) \pmod{N} + 1$. Then we define

$$\mathcal{I} : \mathsf{R}(z) \rightarrow \mathsf{R}^N(z) : u(z) = \sum_{k=0}^{\infty} u[k] z^{-k} \mapsto \overline{u}(z) = \sum_{k=0}^{\infty} \overline{u}[k] z^{-k}$$

by means of

$$\left. \begin{array}{rcl@{\qquad}l} \overline{u}_i[k] & = & u[k] & i = i_k \\ \overline{u}_i[k] & = & 0 & i \neq i_k \end{array} \right\} . \tag{6}$$

We also take $\overline{x}_1[0] = x[0]$ and $\overline{x}_i[0] = 0$ for $i \neq 1$. Then it is easy to check that

$$\left. \begin{array}{rcl@{\qquad}l} \overline{y}_{i_k}[k] & = & y[k] & i = i_k \\ \overline{y}_i[k] & = & 0 & i \neq i_k \end{array} \right\} . \tag{7}$$

for all integers $k \geq 1$. In other words, the diagram in Figure 2 commutes, with Π defined by

$$\Pi : \mathsf{R}^N(z) \rightarrow \mathsf{R}(z) : \overline{y}(z) = \sum_{k=0}^{\infty} \overline{y}[k] z^{-k} \mapsto y(z) = \sum_{k=0}^{\infty} y[k] z^{-k}.$$

Denote as $S_2(z)$ the transfer function for $\mathcal{S}_1$ represented in this form. Let $R_2(z) = S_2(z)(I_{N \times N} + S_2(z))^{-1}$ be the complementary sensitivity transfer function. Now we assume $\mathcal{K}$ is periodic with period N as in problem (3.). Denote the transfer function for $\mathcal{K}$ using the staggered representation as $K(z)$.

Now we shall prove the following:

$$\begin{tikzcd}
\mathsf{R}^N(z) \arrow[r, "S_2(z)"] & \mathsf{R}^N(z) \arrow[d, "\Pi"] \\
\mathsf{R}(z) \arrow[u, "\mathcal{I}"] \arrow[r, "\mathcal{S}_1"] & \mathsf{R}(z)
\end{tikzcd}$$

Figure 2: Relationship Between Periodic and Time-Invariant Systems

Theorem 1 *The norm of $R_2(z)$ is equal to the reciprocal of the smallest norm of all destabilizing periodic perturbations of $\mathcal{S}_1$.*

Let $\mathcal{P}$ denote the set of PSISO systems, and let $\mathcal{T}$ denote the set of $N \times N$ transfer function matrices. The proof uses three propositions:

Proposition 1 *The map $G : \mathcal{P} \to \mathcal{T}$ defined by*

$$G(\{A_i\}, \{B_i\}, \{C_i\}, \{D_i\}) \mapsto \overline{C}(sI_{N\times N} - \overline{A})^{-1}\overline{B} + \overline{D},$$

which takes $\mathcal{S}_1$ to $S_2(z)$, is norm-preserving.

Proof: $\mathcal{I}$ is an isometry and Π is a left inverse of $\mathcal{I}$. Therefore $\|\mathcal{S}_1\| \leq \|S_2\|$. The reverse inequality follows from observing that

$$\mathsf{R}^N(z) = \oplus_{i=1}^{N}\left(z^{1-i}\mathcal{I}(\mathsf{R}(z))\right),$$

$$z^i\mathcal{I}(\mathsf{R}(z)) \perp_{\ell_2} z^k\mathcal{I}(\mathsf{R}(z)) \quad \text{for} \quad i \neq k,$$

and that $S_2(z)$ (of course) commutes with multiplication by z^{-1}. ∎

Proposition 2 *$G\mathcal{K}$ destabilizes S_2 if and only if $\mathcal{K}$ destabilizes $\mathcal{S}_1$.*

Proof: We first note that G commutes with feedback interconnections.

(only if) $G\mathcal{K}$ destabilizes S_2 implies the closed loop transfer function is unbounded on some element of one of the subspaces of $\mathsf{R}^N(z)$ of the form $z^i\mathcal{I}(\mathsf{R}(z))$, which means that it is unbounded on $\mathcal{I}(\mathsf{R}(z))$. From Figure 2, $\mathcal{K}$ destabilizes $\mathcal{S}_1$.

(if) The diagram shows that an input for the feedback connection of the periodic systems for which the output is unbounded gives a similar input for the time-invariant representaton. ∎

Proposition 3 *If σ is a singular value of $R_2(e^{j\omega_0})$, there is an element of $G\mathcal{P}$ having norm $1/\sigma$ which destabilizes the closed loop.*

Proof: Suppose σ_i is a singular value of $R_2(e^{j\omega_0})$. Then we can write

$$R_2(e^{j\omega_0}) = V(e^{j\omega_0})\Sigma(e^{j\omega_0})U^H(e^{j\omega_0})$$

with U and V being Hermitian matrices, and Σ being diagonal. Writing

$$V(e^{j\omega_0}) = [v_1 \vdots v_2 \vdots \ldots v_N], \qquad U(e^{j\omega_0}) = [u_1 \vdots u_2 \vdots \ldots u_N],$$

$$\text{and} \quad \Sigma(e^{j\omega_0}) = \operatorname{diag}\{\sigma_1, \sigma_2, \ldots, \sigma_N\},$$

let

$$\Delta_i \triangleq \frac{1}{\sigma_i} u_i v_i^H.$$

It is easy to check that $I - R_2(e^{j\omega_0})\Delta_i$ is singular, and $\|\Delta_i\| = 1/\sigma_i$. However, generally $\Delta_i \notin \mathcal{GP}$. We shall find $\Delta(z) \in \mathcal{GP}$ such that $\Delta(e^{j\omega_0}) = \Delta_i$.

First we note that a straighforward calculation shows that $\mathcal{GP}$ consists of matrices of the form

$$\frac{\begin{bmatrix} C_1A_2A_3\ldots A_NB_1 & C_1A_3\ldots A_NzB_2 & \cdots & C_1z^{N-1}B_N \\ C_2z^{N-1}B_1 & C_2A_1A_3\ldots A_NB_2 & \cdots & C_2A_1z^{N-2}B_N \\ C_3A_2z^{N-2}B_1 & C_3z^{N-1}B_2 & \cdots & C_3A_1A_2z^{N-3}B_N \\ \vdots & \vdots & & \vdots \\ C_NA_2\ldots A_{N-1}zB_1 & C_NA_3\ldots A_{N-1}z^2B_2 & \cdots & C_NA_1\ldots A_{N-1}B_N \end{bmatrix}}{z^N - A_1A_2\ldots A_N}.$$

Now let

$$u_1 = \begin{bmatrix} u_{11} \\ u_{12} \\ \vdots \\ u_{1N} \end{bmatrix} \quad and \quad v_1 = \begin{bmatrix} v_{11} \\ v_{12} \\ \vdots \\ v_{1N} \end{bmatrix}.$$

We define

$$u(z) = \begin{bmatrix} u_{11} \\ u_{12}e^{j\omega_0}z^{-1} \\ \vdots \\ u_{1N}e^{j(1-N)\omega_0}z^{1-N} \end{bmatrix} \quad and \quad v(z) = \begin{bmatrix} v_{11} \\ v_{12}e^{j\omega_0}z^{-1} \\ \vdots \\ v_{1N}e^{j(1-N)\omega_0}z^{1-N} \end{bmatrix}.$$

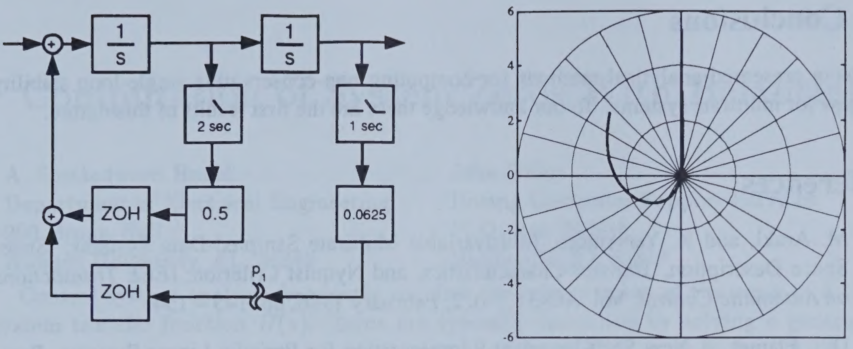
Finally, we let

$$\Delta(z) \triangleq \frac{1}{\sigma_i} u(z) v^H(z)$$

and observe that $\Delta(z) \in \mathcal{GP}$, $\Delta(e^{j\omega_0}) = \Delta_i$, and $\|\Delta(z)\| = 1/\sigma_i$. ∎

Proof of theorem: Calculate $\overline{\sigma} = \|R_2(z)\|$. By Proposition 3 find a corresponding destabilizing feedback which is the staggered version of a periodic system. By Proposition 2 this periodic system destabilizes S_1. By Proposition 1 this periodic perturbation has norm $1/\overline{\sigma}$. Conversely, every periodic destabilizing perurbation of S_1 gives a transfer function which has the same norm and destabilizes $R_2(z)$. ∎

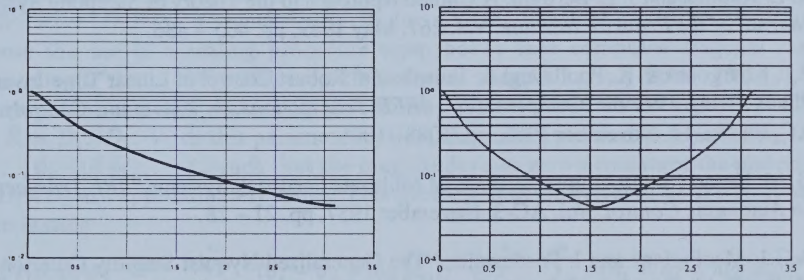
This theorem shows that the representation $S_2(z)$ may be used instead of $S_1(z)$ for the stability analysis of PSISO systems. The dimensions of the matrix $S_2(z)$ are the same as those of $S_1(z)$, but the calculation of S_2 is much easier to automate since no computation is required in finding the coefficient matrices in the state space representation.



Multirate System Example

Polar Plot of Largest Return Ratio Eigenvalue

Figure 3: (a) Block Diagram; (b) Constant Perturbation Margins



Stacked System

Staggered System

Largest Singular Value of Complementary Sensitivity

Figure 4: Periodic Perturbation Margin Computation

5 Example

In this section we demonstrate the relationships between the various margins and computations with an example. The example in Figure 3 depicts a multirate system for which a SISO transfer function does not exist for finding margins at the fast rate point. The accompanying figure shows a polar plot of the largest eigenvalue for the return ratio, from which one can read the gain margin, phase margin, and m-circle radius. Figure 4 shows the largest singular values of the complementary sensitivity for the blocked and stacked representations. Note that the maximum is the same, and that the plot for the staggered system is a doubled-over version of that for the stacked system.

The author would like to acknowledge the assistance of Mr. Reynold C. Wang and his work in preparing this example.

6 Conclusions

We have presented analytical methods for computing non-conservative single-loop stability margins for multirate systems. To our knowledge these are the first results of this nature.

References

[1] M. Araki, and K. Yamamoto, Multivariable Multirate Sampled-Data Systems: State-Space Description, Transfer Characteristics, and Nyquist Criterion, *IEEE Transactions on Automatic Control*, Vol. AC-31, No. 2, February 1986, pp. 145 – 154.

[2] D.S. Flamm, A New Shift-Invariant Representation for Periodic Linear Systems, *Proc. American Control Conference*, May 1990, San Diego, CA, pp. 1510-1515.

[3] B. A. Francis and T. T. Georgiou, Stability Theory for Linear Time-Invariant Plants with Periodic Digital Controllers, *IEEE Transactions on Automatic Control*, Vol. 33, No. 9, September 1988, pp. 820 – 832.

[4] R. E. Kalman and J. E. Bertram, A Unified Approach to the Theory of Sampling Systems, *Journal of the Franklin Institute*, Vol. 267, May 1959, pp. 405 – 436.

[5] P. P. Khargonekar, K. Poolla and A. Tanenbaum, Robust Control of Linear Time-Invariant Plants Using Periodic Compensation, *IEEE Transactions on Automatic Control*, Vol. AC-30, No. 11, November 1985, pp. 1088– 1096.

[6] G. M. Kranc, "Input-Output Analysis of Multirate Feedback Systems," *IRE Transactions on Automatic Control*, Vol. AC-3, November 1957, pp. 21 – 28.

[7] A.G.J. MacFarlane and I. Postlewaite, "The Generalized Nyquist Stability Criterion and Multivariable Root Loci," *Int. J. Control*, Vol. 25, Jan. 1977, pp. 81 - 127.

[8] P.M. Thompson, "Gain and Phase Margins of Multi-rate Sampled-data Feedback Systems," *Int. J. Control*, Vol. 44, No. 3, 1986, pp. 833-846.

Computation of System Zeros With Balancing*

A. Scottedward Hodel
Department of Electrical Engineering
200 Broun Hall
Auburn University, AL, 36849

John Tiller
Boeing Computer Support Services
P. O. Box 240010
Huntsville, AL 35812

Control system analysis frequently requires the computation of the zeros of a linear system transfer function $H(s)$. Zeros are typically computed by solving a generalized eigenvalue problem (GEP) $\bar{A}x = \lambda \bar{E}x$, $\bar{A} = \begin{bmatrix} -A & -B \\ C & D \end{bmatrix}$ and $\bar{E} = \begin{bmatrix} I_n & 0 \\ 0 & 0_{p\times m} \end{bmatrix}$, conformably partitioned. We shall refer to a GEP of this structure as a zero-computation generalized eigenvalue problem (ZGEP). Emami-Naeini and Van Dooren [2] present an algorithm for the solution of the ZGEP that selects a lower order ZGEP with the the same finite zeros as the original problem by isolating zeros at $s = \infty$.

Direct application of Ward's GEP balancing procedure [4] is inappropriate for the ZGEP, since $D_1 \bar{E} D_2 \neq \text{block diag}(I, 0)$ for general diagonal matrices D_1, D_2. We propose the use of a scaling procedure `zgep_scale` that computes diagonal matrices $D_1 = \text{diag}(2^{\sigma_1}, \ldots, 2^{\sigma_n}, 2^{\psi_1}, \ldots, 2^{\psi_p})$, $D_2 = \text{diag}(2^{-\sigma_1}, \ldots, 2^{-\sigma_n}, 2^{\phi_1}, \ldots, 2^{\phi_m})$. (Other potential ZGEP balancing procedures may be found in [3] and [5].) Observe that $\bar{E} = D_1 \bar{E} D_2$. With this parameterization, the scaling problem is to select $\sigma_1, \ldots, \sigma_n$, $\psi_1, \ldots, \psi_p$ and $\phi_1, \ldots, \phi_m$ such that the magnitudes non-zero elements of the matrix product $D_1 \bar{A} D_2$ are approximately unity. This problem may be restated as the least-squares minimization

$$\begin{aligned} \min_{\sigma,\phi,\psi} f(\sigma,\phi,\psi) \;=\; & \sum_{i,j=1,a_{ij}\neq 0}^{n} (\sigma_i - \sigma_j + \log|a_{ij}|)^2 + \sum_{i=1}^{n} \sum_{j=1,b_{ij}\neq 0}^{m} (\sigma_i + \phi_j + \log|b_{ij}|)^2 \qquad (1) \\ & + \sum_{i=1}^{p} \sum_{j=1,c_{ij}\neq 0}^{n} (\psi_i - \sigma_j + \log|c_{ij}|)^2 + \sum_{i=1}^{p} \sum_{j=1,d_{ij}\neq 0}^{m} (\psi_i + \phi_j + \log|d_{ij}|)^2. \end{aligned}$$

Following the analysis of Ward [4], we equate the partial derivatives of equation (1) to zero to obtain the (possibly singular) consistent set of equations

$$\begin{bmatrix} F_1 & B_\omega & -C_\omega{}^T \\ B_\omega{}^T & F_2 & D_\omega{}^T \\ -C_\omega & D_\omega & F_3 \end{bmatrix} \begin{bmatrix} \sigma \\ \phi \\ \psi \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} \qquad (2)$$

where B_ω, C_ω and D_ω are the incidence matrices of B, C, and D, respectively,

$$\begin{aligned} (F_1)_{ij} &= \sum_{k=1,k\neq i}^{n} (\delta(a_{ki}) + \delta(a_{ik})) + \sum_{k=1}^{m} \delta(b_{ik}) + \sum_{k=1}^{p} \delta(c_{ki}) \\ (F_2)_{ij} &= \sum_{k=1}^{n} \delta(b_{ki}) + \sum_{k=1}^{p} \delta(d_{ik}) \\ (F_3)_{ij} &= \sum_{k=1}^{n} \delta(c_{ik}) + \sum_{k=1}^{m} \delta(d_{ik}) \end{aligned}$$

---

*This work was supported in part by the NASA-ASEE Summer Faculty Fellowship Program and in part by NASA Contract No. NAS8-37200. (c) Copyright 1991, The Boeing Co., All Rights Reserved.

where $\delta(x) = 1$ if $x \neq 0$, else $\delta(x) = 0$, and

$$\begin{aligned}
z_i^{(1)}, i = 1, \ldots, n &= \sum_{j=1, j\neq i}^{n} (\log|a_{ji}| - \log|a_{ij}|) - \sum_{j=1}^{m} \log|b_{ij}| + \sum_{j=1}^{p} \log|c_{ji}| \\
z_i^{(2)}, i = 1, \ldots, m &= -\sum_{j=1}^{n} \log|b_{ji}| - \sum_{j=1}^{p} \log|d_{ji}| \\
z_i^{(3)}, i = 1, \ldots, p &= -\sum_{j=1}^{n} \log|c_{ij}| - \sum_{j=1}^{m} \log|d_{ij}|
\end{aligned}$$

As in [4], optimal scaling parameters of equation (2) may be obtained through a generalized conjugate gradient iteration [1].

Example We illustrate the importance of preconditioning the ZGEP in the following seventh order SISO dynamic system based upon the space shuttle main engine oxidizer-preburner valve dynamics. The nonzero system coefficients are $a_{11} = -7000$, $a_{12} = -2.5 \times 10^7$, $a_{17} = -1.82943 \times 10^9$, $a_{21} = 1$, $a_{32} = 6.4 \times 10^5$, $a_{33} = -2,240$, $a_{34} = -a_{32}$, $a_{43} = 1$, $a_{54} = 9.03934$, $a_{65} = 225,449$, $a_{66} = -3,000$,$a_{67} = -2.25 \times 10^6$, $a_{76} = 1$, $b_{11} = 1.44813 \times 10^8$, $c_{15} = 1.26582$. The two finite system zeros are (to 4 significant digits) $s = 1500 \pm j2.7847 \times 10^{-5}$. Application of [2] without preconditioning yields six finite computed zeros at -1904, $(-1500 \pm j8.744 \cdot 10^4)$, -336.2 and two zeros at the origin. EISPACK routines `qzhes`, `qzit` and `qzval` without preconditioning yields zeros at $-5.468 \cdot 10^5$, $(2.688 \pm j4.720) \cdot 10^5$, $(9.913 \pm j9.359) \cdot 10^7$, and two zeros at -1500. Application of `zgep_scale` in tandem with [2] or EISPACK also yields the correct zeros at -1500. The `zgep_scale` scaled system coefficients are $a_{11} = -7000$, $a_{12} = -762.9$, $a_{17} = -1744.7$, $a_{21} = 32,768$, $a_{32} = 78.125$, $a_{33} = -2240$, $a_{34} = -156.25$, $a_{43} = 4096$, $a_{54} = 72.31472$, $a_{65} = 27.52$, $a_{66} = -3000$, $a_{67} = -34.33$, $a_{76} = 65,536$, $b_{11} = 1.0789$ and $c_{15} = 0.6329$.

References

[1] Paul Concus, Gene H. Golub, and Dianne P. O'Leary. A generalized conjugate gradient method for the numerical solution of elliptic partial differential equations. In J. R. Bunch and D. J. Rose, editors, *Sparse Matrix Computations*, pages 309–332. Academic Press, 1976.

[2] A. Emami-Naeini and P. Van Dooren. Computation of zeros of linear multivariable systems. *Automatica*, 18(4):415–430, 1982.

[3] A. Emami-Naeini and P. Van Dooren. On computation of transmission zeros and transfer functions. In *Proceedings of the 21st IEEE Conference on Decision and Control*, pages 51–55, Orlando, Fl., Dec.8-10 1982.

[4] Robert C. Ward. Balancing the generalized eigenvalue problem. *Siam J. Sci. Stat. Comput.*, 2(2):141–152, June 1981.

[5] T. W. C. Williams. Obtaining minimal Gerschgorin discs by scaling the states. *International Journal of Control*, 42(5):1155–1173, 1985.

ON THE RECOGNITION OF STRONGLY CHORDAL GRAPHS

Radhakrishnan Sridhar‡ and Narayan Chandrasekharan†

‡ School of Electrical Engineering and Computer Science
University of Oklahoma
Norman, OK 73019.

† Department of Computer Science
University of Central Florida
Orlando, FL 32169.

ABSTRACT

Given a graph G with n vertices and m edges we present an $O(n^2)$ algorithm to determine if G is strongly chordal. Our recognition algorithm for strongly chordal graphs improves the previous result of Paige and Tarjan which runs in $O(n^2 logn)$ time. β-acyclic hypergraphs are precisely strongly chordal graphs which has applications in databases, VLSI design, networks, sparse matrix computations, and expert systems. Many NP-Complete problems can be solved in polynomial time when the input graph is restricted to strongly chordal graphs.

1. INTRODUCTION

A simple, loopless, and undirected graph is *chordal* if every cycle of length at least four contains a chord, i.e., an edge between two vertices that are not consecutive in the cycle. A *strongly chordal graph* is a chordal graph that in every even cycle with at least six nodes contains a strong chord (i.e., a chord joining two nodes with an odd distance in the cycle). Chordal graphs are an important class of perfect graphs [15]. The class of graphs Ψ which contains strongly chordal and chordal graphs have been shown to be closely related to the acyclicity of the hypergraph, formed using the maximal cliques of the graph in Ψ [1]. Graphs in Ψ arise in several important applications, which include the following.

(a). Relational database design and query evaluation [3, 4, 8].

(b). Computing solutions of sparse system of linear equations [2, 9].

(c). Probabilistic expert systems [16].

(d). Reliability of communication networks in the presence of constrained line and site failures [11, 12, 19].

(e). VLSI design layout [18].

The *closed neighborhood* of a vertex v denoted $N[v]$, is the set consisting of v together with all vertices which are adjacent to v. A *perfect elimination ordering* of a graph $G = (V, E)$ is an ordering $(<)$ $v_1, v_2, ..., v_n$ of V with the property that for each i, j, and l, if $i < j$, $i < l$, and $v_l, v_j \in N[v_i]$, then $v_l \in N[v_j]$. A graph is *chordal* if and only if it admits a perfect elimination ordering (see [15]). Strongly chordal graphs are defined in terms of a stronger ordering condition. A *strong elimination ordering* of a graph $G = (V, E)$ is an ordering $(<)$ $v_1, v_2, ..., v_n$ of V with the property that for each i, j, k and l, if $i < j$, $k < l$, $v_l \in N[v_i]$, and $v_k \in N[v_j]$, then $v_l \in N[v_j]$ [9]. A graph is strongly chordal if and only if it admits a strong elimination ordering.

Given a strong elimination ordering, linear-time sequential algorithms for several domination problems have been given in the past [6, 10]. Many domination problems are NP-complete on chordal graphs [5, 13]. Paige and Tarjan [17] present an $O(n^2 logn)$ algorithm to determine the strong elimination ordering thus providing a recognition algorithm for strongly chordal graphs. Now, several domination problems can be solved in $O(n^2 logn)$ time instead of $O(n^3)$ time as stated in [6, 10].

In this paper, we present an $O(n^2)$ algorithm to recognize strongly chordal graphs, thus improving the result of Paige and Tarjan [17]. Also, recently Cheung and Zhu [7] present an interesting algorithm to determine different kinds of beta-cycles. It is previously known that strongly chordal graphs are beta-acyclic hypergraphs [1]. Hence the algorithm in [7] can be used to recognize strongly chordal graphs. The algorithm of Cheung and Zhu has a complexity of $O(n. |E|)$, where n is the number of vertices and $|E|$ is the number of pairwise non-empty intersection between maximal cliques of the strongly chordal graph. In the worst case $|E| = n^2$, hence the algorithm in [7] has a time complexity of $O(n^3)$ in the worst case. In this paper we will present an algorithm which has a better time-complexity than the algorithms in [7, 17]. We strongly believe that our algorithm can be easily parallelized and hence an NC algorithm for the recognition of strongly chordal graphs can be obtained.

Our paper is organized along the following lines. In section 2 we present an algorithm to construct the intersection graph of a strongly chordal graph. In section 3 we derive several important properties of the intersection graph of a strongly chordal graph. Based on the characterizations developed in section 3 an $O(n^2)$ algorithm is presented in section 4 to recognize strongly chordal graphs.

2. THE INTERSECTION GRAPH OF A STRONGLY CHORDAL GRAPH

In this section, we will define the intersection graph $I(H)$ of the maximal cliques H of a strongly chordal graph G and present a $O(n^2)$ time sequential and algorithm to construct the intersection graph. We next derive several properties of the intersection graph of the strongly chordal graph.

Definition 2.1 [Intersection Graph]: Let H be the maximal cliques $\{S_1, S_2, ..., S_r\}$, $1 \leq r \leq n$, of a chordal graph G. We can treat H as a hypergraph by treating each maximal clique in H as a hyperedge. The *intersection graph* $I(H)$ of H is a graph containing edges $S_i \in H$ as nodes and edges (S_i, S_j) labeled $l(S_i \cap S_j)$, if $S_i \cap S_j \neq \emptyset$. The size of an edge label $l_{ij} = S_i \cap S_j$ is $|S_i \cap S_j|$.■

The following equivalence was proved by D'Atri and Moscarini [1].

Theorem 2.2: A graph G is strongly chordal if and only if the hypergraph formed using the maximal cliques H of G is β-acyclic.■

Definition 2.3 [Hypergraph Acyclicity]:

A hypergraph H is β-*acyclic* [8] if it does not contain the following sequence called as a **β-cycle**: A sequence $(S_1, ..., S_m, S_{m+1})$ of edges is a β-cycle if $X = S_1 \cap ... \cap S_m$, and S_i' is the set difference $S_i - X$ $(1 \leq i \leq m)$, then $(S_i', ..., S_m', S_{m+1}')$ is a *pure cycle*. A sequence $S = (S_1, ..., S_m, S_{m+1})$ be a sequence of distinct edges and $S_{m+1} = S_1$. Let us call S_i and S_{i+1} *neighbors* $(1 \leq i \leq m)$. The sequence S is a *pure cycle* if $m \geq 3$ (i.e., at least three edges are involved) and if whenever $i \neq j$, then $S_i \cap S_j$ is non-empty if and only if S_i and S_j are neighbors.■

The following combinatorial lemma holds for chordal graphs (see [15]; page 98).

Lemma 2.4: A chordal graph on n vertices has at most n maximal cliques, with equality if and only if the graph has no edges.■

Lemma 2.5: The sum of the sizes of the edge labels in the intersection graph $I(H)$ formed using the maximal cliques H of a chordal graph G with n-vertices is $O(n^2)$.

Proof: A vertex v in the edge label $l(x, y)$ of the intersection graph $I(H)$ is connected to vertices in x and y. This means that v is connected to at least two different vertices, one in x and the other in y assuming that $x \neq y$. The vertex v is adjacent to at most n vertices in G. Hence, each vertex v can be in at most n edges in $I(H)$. Hence the Lemma.■

We now present a $O(n^2)$ algorithm for constructing the intersection graph $I(H)$ given the maximal cliques H represented as vertex-clique incidence lists.

```
Algorithm Construct_Intersection_Graph;
Input:    Maximal cliques $H$ represented as vertex-clique incidence list $M$.
          $M(i)$ is the list for clique $S_i$. Let $M(i).j$ represent the number of
          the $j$th clique in $M(i)$.
Output:   Intersection graph $I(H)$ represented as an incidence list $L$.
          $L(i)$ is the pairs $(i_1, l_1), ..., (i_k, l_k)$, where $l_j = S_i \cap S_{i_j} \neq \emptyset$.
Data structures:
(1) An $n \times n$ uninitialized matrix $R$ with columns and rows labeled with clique numbers.
(2) $R(i, j)$ points to a set which stores $S_i \cap S_j \neq \emptyset$.
(3) A 'Flag' field in each $R(i, j)$ call it Flag($i$, $j$) which is uninitialized.
(4) A multi-set $P(i)$ for each clique $S_i$. $P(i).j$ means that the $j$th element of $P(i)$, and
    $S_i \cap S_{P(i).j} \neq \emptyset$.

Begin
1.  For $i$ := 1 to $n$ do
2.      For $j$ := 1 to $|M(i)|$ do
3.          For $k$ := 1 to $|M(i)|$ do
4.              If ($j \neq k$) then
                Begin
5.                  Add vertex $v_i$ into the set in $R(M(i).j, M(i).k)$;
6.                  Insert the number $M(i).k$ into the multi-set $P(M(i).j)$;
7.                  Flag($M(i).j$, $M(i).k$) := 0;
                End;
8.  For $i$ := 1 to Number_of_cliques do
9.      For $j$ := 1 to $|P(i)|$ do
10.     If Flag($i$, $P(i).j$) = 0 then
        Begin
11.         Add the pair ($P(i).j$, the set pointed by $R(i, P(i).j)$) into $L(i)$;
12.         Flag($i$, $P(i).j$) = 1;
        End;
End.
```

We will show that the above algorithm has a $O(n^2)$ time complexity. For each vertex v_i we need to generate edges with label v_i among all cliques in $M(i)$, i.e. the cliques which contain vertex v_i. This is done in steps 2-7. Clearly, from the proof of Lemma 2.5 since v_i can be in at most n edges of $I(H)$, the steps 2-7 is executed at most n times for each vertex v_i. Thus steps 1-7 take $O(n^2)$ time to complete. Using similar argument we can show that the cardinality of each $P(i)$ is at most n. Thus, steps 8-12 can also be completed in $O(n^2)$ time. The total time taken to construct the intersection graph given a set of maximal cliques represented as vertex-clique incidence lists is $O(n^2)$.

3. PROPERTIES OF THE INTERSECTION GRAPH OF A STRONGLY CHORDAL GRAPH

In this section we derive several properties of strongly chordal graphs based on the intersection graph $I(H)$ of its maximal cliques H.

Lemma 3.1: The *intersection* graph $I(H)$ for a set of maximal cliques H of the graph G is chordal, if G strongly chordal.

Proof: When graph G is strongly chordal we know from Theorem 2.1 that H is β-acyclic. We will show by contradiction that $I(H)$ is chordal. Let H be β-acyclic and $I(H)$ not chordal. Consider the sequence $(S_1, x_1, S_2, x_2, S_3, x_3, S_4, x_4, S_1)$ in $I(H)$. The nodes x_1, x_2, x_3, and x_4 are all distinct (otherwise there would be a chord connecting opposite vertices). The sequence is β-cycle, which means that H contains a β-cycle, a contradiction. Therefore, $I(H)$ is chordal when G strongly chordal.■

Lemma 3.2: If H is β-acyclic, then, every triangle (S_i, S_j, S_k) in $I(H)$ satisfies the condition $S_p \cap S_q \subseteq S_p \cap S_r$ and $S_p \cap S_q \subseteq S_q \cap S_r$ for some $p \neq q \neq r$ in $[i, j, k]$.

Proof: Let $S_q \cap S_q = r_1, S_q \cap S_r = r_2, S_r \cap S_q = r_3$, and H be β-acyclic. If $r_1 \cap r_2 \cap r_3 = \emptyset$, then, we have a β-cycle a contradiction to the assumption. If $r_1 \cap r_2 \cap r_3 = x$ and $x \subset r_i$, $i = 1, 2, 3$, then we have a β-cycle by Definition 2.3, a contradiction. The case where, $(r_1 = r_2) \cap r_3 = \emptyset$ does not exists, and either $r_3 = r_1$ or the containment stated in the Lemma holds.■

The other properties of the intersection graph are stated in the following propositions.

Proposition 3.3: Let H_i^* be a maximal clique of $I(H)$ of a strongly chordal graph. Let l_i be an edge label whose size is minimum among all edge labels in H_i^*. The label l_i is called the **minimum edge label**. The label l_i is contained in each vertex of H_i^*.

Proof: Since l_i is the edge label whose size is minimum it is contained in every other edge in H_i^* from Lemma 3.2. This implies l_i is in every vertex of H_i^* (i.e., adjacent to all vertices of G in H_i^*).■

The following proposition states that each maximal clique of the intersection graph of a strongly chordal graph contains at most one minimum edge label.

Proposition 3.4: Let H_i^* be a maximal clique of $I(H)$ of a strongly chordal graph. H_i^* does not contain two minimum edge labels l_i and l_j with $l_i \cap l_j \neq l_i$.

Proof: If it contains two minimum edge labels l_i and l_j then by Proposition 3.3 l_i and l_j should be in all the vertices of H_i^*. This implies that $l_i \cup l_j$ is the minimum edge label. Hence the proposition.■

Proposition 3.5: Let H_i^* and H_j^* be two maximal cliques of $I(H)$ of a strongly chordal graph with minimum edge labels l_i and l_j, respectively. We have $l_i \cap l_j = \emptyset$.

Proof: If $l_i \cap l_j \neq \emptyset$, then, every vertex in H_j^* is connected to every vertex in H_i^* (from Proposition 3.3) which implies $H_i^* = H_j^*$, a contradiction.■

The following lemma is a set of all necessary conditions on $I(H)$ for a graph to be strongly chordal. This lemma is a constructive one used in our algorithm.

Lemma 3.6: Let $I(H)$ be the intersection graph of a strongly chordal graph. Let $\{H_1^*, ..., H_k^*\}$ be the maximal cliques of $I(H)$, with minimum edge labels $L = \{l_1, ..., l_k\}$, respectively. The intersection graph $I(H)$ satisfies the following conditions:

1. Lemma 3.1
2. Lemma 3.2
3. Proposition 3.3
4. Proposition 3.4
5. For each $1 \leq i \leq k$, removal of l_i from the label of each edge in H_i^* should leave H_i^* acyclic (i.e a graph without cycles).

Proof: Proof that it satisfies 1-4 is clear from proofs given earlier. Now for condition 5 we can clearly see for otherwise, there exists a β-cycle in $I(H)$, contradicting the assumption that $I(H)$ is the intersection graph of a strongly chordal graph.■

We will now state our main characterization theorem for strongly chordal graphs.

Theorem 3.7: Let G be a graph and let $I(H)$ be its intersection graph. The graph G is strongly chordal if and only if it satisfies the conditions in Lemma 3.6.

Proof: If G is strongly chordal then, from proof of Lemma 3.6 the conditions in Lemma 3.6 will be satisfied. We will now show that if $I(H)$ the intersection graph of a graph G satisfies the conditions stated in Lemma 3.6 then, the graph G is strongly chordal. The proof is by contradiction. Let us assume that a graph G with its intersection graph $I(H)$ satisfies conditions in Lemma 3.6 but $I(H)$ contains a β-cycle. Let $C = \{S_1, S_2, ..., S_i\}$ be the vertices of $I(H)$ which forms the β-cycle. If C is a clique in $I(H)$ then clearly vertices in C does not form a β-cycle since it satisfies condition 5 of Lemma 3.6.

Assume that vertices in C do not form a clique in $I(H)$. Now vertices in C induces two or more maximal cliques in $I(H)$. Note that if two maximal cliques in $I(H)$ share an edge then the minimum edge label for both the maximal cliques is not on the edge being shared from Lemma 3.2. The vertices in C form a β-cycle if for some maximal clique $C_1 = \{S_j, S_{j+1}, ..., S_k\}$ with $C_1 \in C$, and **no** $S_a \cap S_{a+1}$ should be in $S_b \cap S_{b+1}$, with $j \leq a \neq b \leq k$. Call the chain C_1 satisfying the above property as a **β-chain**. This is not possible since, clique C_1 shares edges with other cliques induced by vertices in C (from condition 2 of Lemma 3.6) at least one $S_a \cap S_{a+1} = x$, where x is the minimum edge label (only one minimum edge label from condition 4 of Lemma 3.6) for the maximal clique C_1 is contained in $S_b \cap S_{b+1}$ for some $j \leq b \neq a \leq k$. Hence C_1 is not a β-chain. So C is not a β-cycle. For example let $Q = \{S_1, S_2, S_3, S_4\}$. Let $S_1 \cap S_2 \cap S_3 = x$ and $S_1 \cap S_2 \cap S_4 = y$. Now, $x \cup y \subseteq S_1 \cap S_2$ from Lemma 3.2 and

either, $S_1 \cap S_3 = x$ or $S_2 \cap S_3 = x$, and
either, $S_1 \cap S_4 = y$ or $S_2 \cap S_4 = y$.

Clearly, Q above is not a β-chain. Hence the theorem.■

4. RECOGNITION ALGORITHM

In this section we present the recognition algorithm based on the characterizations developed in section 3. We will assume that the input graph G is chordal if not clearly G is not strongly chordal.

```
Algorithm RECOG;
Input:   A chordal graph $G$ with $n$ vertices and $m$ edges.
Output:  'yes' if $G$ is strongly chordal, 'no' otherwise.
begin
1.   determine the maximal cliques of $G$ and represent them as vertex-clique incidence lists;
2.   use algorithm Construct_Intersection_Graph and construct $I(H)$;
3.   if $I(H)$ is not chordal then exit ('no')
4.   determine $\{H_1^*, ..., H_k^*\}$ the maximal cliques of $I(H)$;
5.   for $i := 1$ to $k$ do
       begin
6.         determine the minimum edge label $l_i$ of the maximal clique $H_i^*$;
7.         if two minimum edge labels exists then exit ('no')
8.         remove vertices in $l_i$ from labels of edges in $H_i^*$ and call the resulting graph as $h$.
9.         if there exists an edge in $H_i^*$ from which $l_i$
                 was not removed or $h$ contains a cycle
           then exit ('no')
       end;
10.  exit ('yes');
end.
```

The algorithm RECOG is correct and is established using Theorem 3.7. The time complexity of the algorithm RECOG is $O(n^2)$ and is estimated as follows. Gavril [14] provided a linear time sequential algorithm to determine the maximal cliques of a chordal graph and to recognize a chordal graph. Since a strongly chordal graph is a chordal graph, we can use the same algorithm to determine the maximal cliques of a strongly chordal graph in linear time. Hence, step 1 and steps 3-4 can be executed in linear-time. From section 2 we see that step 2 requires $O(n^2)$ time. Step 5-7 take linear-time since it has been shown in (see [15]; page 98) that the sum of the size of all maximal clique of a chordal graph with n vertices and m edges is $O(n+m)$. Step 8 takes overall (total time taken when the algorithm terminates) $O(n^2)$ time from Lemma 2.5. Step 9 can be executed in linear time in the size of the maximal clique using the standard depth first search algorithm, hence overall in linear-time.

Theorem 4.1: Given a graph G with n vertices and m edges the algorithm RECOG can determine if G is strongly chordal in $O(n^2)$ time.■

References

1. Alessandro, D'Atri and Marina, Moscarini, ''On Hypergraph Acyclicity and Graph Chordality,'' *IPL*, vol. 29, no. 1, November 1988.
2. Anstee, R.P. and Farber, M., ''Characterizations of Totally Balanced Matrices,'' *Journal of Algorithms*, vol. 5, no. 1, pp. 215-230, 1984.
3. Arnborg, S., ''On the Complexity of Multivariable Query Evaluation,'' *FOA Rapport C20292-D8, National Defence Research Institute, Stockholm, Sweden*, 1979.
4. Beeri., Cartiel, Fagin., Ronald, Maier., David, and Yannakakis., Mihalis, ''On the Desirability of Acyclic Database Schemes,'' *JACM.*, vol. 30, no. 3, pp. 479-513, July 1983.
5. Booth, K.S. and Johnson, J.H., ''Dominating Sets In Chordal Graphs,'' *SIAM J. Comput.*, vol. 11, pp. 191-199, 1982.
6. Chang, G.J., ''Labeling Algorithms For Dominating Problems In Sun-Free Chordal Graphs,'' *Disc. Appl. Math.*, vol. 22, pp. 21-34, 1988/89.
7. Cheung, To-Yat and Zhu, Y., ''Recognizing different Types of Beta-Cycles in a Database Scheme,'' *Theoretical Computer Science*, vol. 81, pp. 295-304, 1991.
8. Fagin., Ronald, ''Degrees of Acyclicity of Hypergraphs and Relational Database Schemes,'' *JACM.*, vol. 30, no. 3, pp. 514-550, July 1983.
9. Farber, M., ''Characterizations of Strongly Chordal Graphs,'' *Discrete Mathematics*, vol. 43, no. 1, pp. 173-189, 1983.
10. Farber, M., ''Domination, Independent Domination, and Duality In Strongly Chordal Graphs,'' *Discrete Appl. Math.*, vol. 7, no. 2, pp. 115-130, 1984.
11. Farley, A.M., ''Networks Immune to Isolated Failures,'' *Networks*, vol. 11, pp. 255-268, 1981.
12. Farley, A.M., ''Networks Immune to Line Failures,'' *Networks*, vol. 12, pp. 393-403, 1982.
13. Garey, M.R. and Johnson, D.S., ''Computers and Intractability: A Guide to the Theory of NP-Completeness,'' *W.H. Freeman and Company, San Fransisco*, 1979.
14. Gavril, F., ''Algorithms For Minimum Coloring, Maximum Clique, Minimum Covering By Cliques, and Maximum Independent Set Of A Chordal Graph,'' *SIAM J. Comput.*, vol. 1, pp. 180-187, 1972.

15. Golumbic, M.C., ''Algorithmic Graph Theory and Perfect Graphs,'' *Academic Press, New York*, 1980.

16. Lauritzen, S.L. and Spiegelhalter, D.J., ''Local Computations with Probabilities on Graph Structures and their Applications to Expert Systems,'' *Journal of Royal Statistical Society*, 1988.

17. Paige, Robert and Tarjan, R.E., ''Three Partition Refinement Algorithms,'' *Siam J. Comput.*, vol. 16, no. 6, pp. 973-989, December 1987.

18. Soumyanath, K. and Deogun, J.S., ''On the Bisection Width of partial-K-trees,'' *Presented at the 20th Southeastern Conf. on Combinatorics, Graph Theory and Computing, Boca Raton, Florida*, Feb. 1989.

19. Wald, J.A. and Colbourn, C.J., ''Steiner Trees, partial-2-trees, and Minimum IFI Networks,'' *Networks*, vol. 13, pp. 159-167, 1983.

AN $O(n \log n)$ ALGORITHM FOR FINDING A DEPTH-FIRST SEARCH TREE IN A PERMUTATION GRAPH

C. Rhee
Department of Mathematics and Computer Science
Eastern Kentucky University, Richmond, KY 40475

Y. Liang
Department of Computer Science
Indiana Purdue University, Fort Wayne, IN 46825

S. K. Dhall and S. Lakshmivarahan
School of Electrical Engineering and Computer Science
University of Oklahoma, Norman, Oklahoma 73019

1. Introduction

Let T be a spanning tree of a connected graph $G = (V, E)$ rooted at an arbitrary node. Then T is called a *depth-first search (DFS)* tree if for each edge $e \in E$, one of the two vertices incident with e is an ancestor of the other in T. It is well-known that the problem of finding a DFS tree for a general graph can be solved in $O(n^2)$ time [2]. In this paper, we develop an $O(n \log n)$ time algorithm for DFS tree in permutation graphs. Let $<n> = \{1, 2, \cdots, n\}$ and $\pi = [\pi_1 \pi_2 \cdots \pi_n]$ be a permutation on $<n>$. Then we construct an undirected graph $G[\pi] = (<n>, E)$ such that $ij \in E$ iff $(i-j)(\pi_i^{-1} - \pi_j^{-1}) < 0$, where π_i^{-1} denotes the position of number i in π. An undirected graph G is a *permutation* graph, if there exists a π such that G is isomorphic to $G[\pi]$ [1]. We, therefore, assume that a permutation graph is represented by the corresponding π. A permutation graph can also be visualized by its corresponding matching diagram constructed as follows. The matching diagram consists of two horizontal parallel channels, named top channel and bottom channel. We put numbers $1, 2, \cdots, n$ on the top channel, in this order, from left to right, and put numbers $\pi_1, \pi_2, \cdots, \pi_n$ on the bottom channel in the same way. Finally, for each i, draw a straight line joining the two i's on the two channels [1]. Fig 1 shows the matching diagram of a permutation graph. We will refer to this figure to illustrate each step of our algorithm.

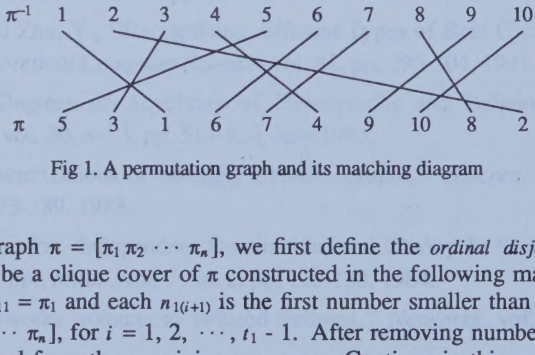


Fig 1. A permutation graph and its matching diagram

2. The algorithm

For a permutation graph $\pi = [\pi_1 \pi_2 \cdots \pi_n]$, we first define the *ordinal disjoint clique cover* $CC(\pi) = (C_1, C_2, \cdots, C_q)$ to be a clique cover of π constructed in the following manner. Let $C_1 = (n_{11}, n_{12}, \cdots, n_{1t_1})$, where $n_{11} = \pi_1$ and each $n_{1(i+1)}$ is the first number smaller than n_{1i}, going from left to right on $\pi = [\pi_1 \pi_2 \cdots \pi_n]$, for $i = 1, 2, \cdots, t_1 - 1$. After removing numbers in C_1 from π, C_2 can be similarly obtained from the remaining sequence. Continue in this manner to get $C_3, C_4, \cdots, C_q$ until all numbers are exhausted. It is clear that $\{C_1, C_2, \cdots, C_q\}$ is a partition of $\{\pi_1, \pi_2, \cdots, \pi_n\}$ and each C_i is a maximal (not necessarily maximum) clique in the subgraph induced by the nodes in $\sum_{j=i}^{q} C_j$. We assume that $CC(\pi)$ and each member of $CC(\pi)$ are all ordered sets. That is, the numbers in each C_i are arranged in the same order as they appear in π and all such cliques appear in $CC(\pi)$ in increasing order of their indices. For example, in Fig 1, $CC(\pi) =$

$(C_1, C_2, C_3, C_4, C_5)$ with $C_1 = (5, 3, 1)$, $C_2 = (6, 4, 2)$, $C_3 = (7)$, $C_4 = (9, 8)$, and $C_5 = (10)$. The computation of $CC(\pi)$ is easily accomplished in $O(n \log n)$ time using binary search.

For each C_i in $CC(\pi)$, let $l(i)$ and $r(i)$ denote the largest number and the smallest number in C_i, respectively. In Fig 1, $C_1 = (5, 3, 1)$, hence $l(1) = 5$ and $r(1) = 1$. For each C_i, let $l{-}adj(i)$ be the largest number j, if any, in C_k, $k < i$, where C_k is the largest-numbered clique containing the line j such that j intersects $l(i)$. If no such j exists, we let $l{-}adj(i) = nil$. Similarly we define $r{-}adj(i)$ to be the smallest number j, if any, in C_k, $k < i$, where C_k is the largest-numbered clique containing the line j such that j intersects $r(i)$. If no such j exists, we let $r{-}adj(i) = nil$. In Fig 1, we have $l{-}adj(1) = nil$, $r{-}adj(1) = nil$, $l{-}adj(2) = nil$, $r{-}adj(2) = 3$, $l{-}adj(3) = 4$, $r{-}adj(3) = nil$, $l{-}adj(4) = 2$, $r{-}adj(4) = nil$, $l{-}adj(5) = 8$, and $r{-}adj(5) = nil$. For each i in π, let $ioq(i)$ be the index of the clique in $CC(\pi)$ to which node i belongs. Then we define $C{-}adj(i)$ to be either $C_{ioq(l-adj(i))}$ or $C_{ioq(r-adj(i))}$, whichever has the larger index. So, $C{-}adj(1) = nil$, $C{-}adj(2) = 1$, $C{-}adj(3) = 2$, $C{-}adj(4) = 2$, and $C{-}adj(5) = 4$. Note that $C{-}adj(i) \neq nil$, $2 \leq i \leq q$, since the graph π is connected.

Lemma 2.1: All $l{-}adj(i)$'s, $r{-}adj(i)$'s and $C{-}adj(i)$'s can be obtained in $O(n)$ time once $CC(\pi)$ is available.

We now construct DFS tree T by computing the parent function, $p(i)$, for each node i in π. For this, we visit each clique in $CC(\pi) = (C_1, C_2, \cdots, C_q)$ in this order starting at C_1. Let $C_1 = (n_{11}, n_{12}, \cdots, n_{1t_1})$. Then all nodes in C_1 will be linked linearly from right to left with n_{11} being the root of T. That is, $p(n_{12}) = n_{11}$, $p(n_{13}) = n_{12}$, $\cdots$, $p(n_{1t_1}) = n_{1(t_1-1)}$. (We call this type of linked list the *backward* chain of the clique. The *forward* one is then the linked list constructed in the other direction.) Then we simply add this chain to T. Now, assuming that we have visited up to C_{i-1} and created a partial DFS tree T accordingly, we show how to augment T with C_i. We first check clique $C_s = C{-}adj(i)$. Two cases will be considered. Firstly, suppose that the chain of C_s is backward. Then check $l{-}adj(i)$. If it is in C_s, then link the nodes in C_i backward and connect the head to $r(s)$. If C_s has only $r{-}adj(i)$, then we link the nodes of C_i forward and connect its head to $r{-}adj(i)$. Secondly, if the chain for C_s is forward, then we need to look at $r{-}adj(i)$. If C_s contains $r{-}adj(i)$, then make the chain of C_i forward and connect the head of the chain to $l(s)$. Otherwise, the nodes in C_i will be linked backward with its head pointing to $l{-}adj(i)$. The time required for this procedure is linear since each node is visited at most twice.

We go back to our example to illustrate this procedure. Initially $T = \varnothing$. For $C_1 = (5, 3, 1)$, we simply construct backward chain $5 \leftarrow 3 \leftarrow 1$ with 5 being the root of T. So, $T = \{(1, 3), (3, 5)\}$. Next, $C{-}adj(2) = 1$ since $r{-}adj(2)(=3)$ is in C_1. Since C_1 has backward chain and only $r{-}adj(2)$ is contained in C_1 we link C_2 forward and let $p(2) = 3$. Therefore, $T = \{(1, 3), (3, 5), (6, 4), (4, 2), (2, 3)\}$. For $C_3 = (7)$, simply connect 7 to 4 since $l{-}adj(3) = 4$ and let 7 be the corresponding backward chain. So, add $(7, 4)$ to T. Consider the next clique $C_4 = (9, 8)$. Recall that $l{-}adj(4) = 2$, $r{-}adj(4) = nil$, hence $C{-}adj(4) = 2$. Since the chain for C_2 is forward and only $l{-}adj(4)$ is in C_2, C_4 will have a backward chain which is $9 \leftarrow 8$. So, $T \leftarrow T \cup \{(8, 9), (9, 2)\}$. Finally, for $C_5 = (10)$, connect 10 to 8 because $C{-}adj(5) = 4$ and $l{-}adj(5) = 8$. Therefore, our final DFS tree for graph in Fig 1 is $T = \{(1, 3), (3, 5), (6, 4), (4, 2), (2,3), (7, 4), (8, 9), (9, 2), (10, 8)\}$.

Theorem 2.2: The algorithm described above finds a DFS tree correctly in $O(n \log n)$ time.

REFERENCES

[1] M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, 1980.

[2] R. E. Tarjan, Depth-first search and linear graph algorithms, *SIAM J. Comput.* 2 (2) (1972) 146-160.

Finding Hamiltonian Paths in Circular-arc Graphs

Y. D. Liang
Department of Computer Science
Indiana Purdue University at Fort Wayne, IN 46805

C. Rhee
Department of Mathematics, Statistics and Computer Science
Eastern Kentucky University, Richmond, KY 40475

S. K. Dhall and S. Lakshmivarahan
Parallel Processing Institute
School of Electrical Engineering and Computer Science
University of Oklahoma, Norman, OK 73019

1. Introduction

The problem of deciding whether a graph has a Hamiltonian Path (HP) has long been known to be NP-complete. Many attempts have been made to characterize the graphs which contain HPs. The HP problem remains NP-complete for many special graphs. Thus far, polynomial time algorithms for the problem have only been developed for a few of special graphs such as 4-connected planar graphs [2], bipartite permutation graphs [2] and interval graphs [3]. Recently, a new and simpler approach was presented to find HP in interval graphs in [5]. In this paper, we present an $O(n^2 logn)$ algorithm for finding HP in circular-arc graphs.

2. Hamiltonian path

Assume that the input graph is given by a circular-arc model $A = \{A_1, \cdots, A_n\}$. Arc A_i is denoted by $[h[i], t[i]]$, $i = 1, \cdots, n$, where $h[i]$ is called the *head point* of A_i and $t[i]$ is called the *tail point* of A_i. Without loss of generality, assume that A_1 does not properly contain any other arc and all points of n arcs are distinct with no arc covering the whole circle. We sort the $2n$ points of the arcs in **A** such that we encounter the points in increasing order if we start at $h[1]$ and travel along the circle in the clockwise direction. This sorting can be done in $O(n \log n)$. Thus we assume that the $2n$ points, denoted by $c_1, \cdots, c_{2n}$, in **A** are available in this order. Let c_0 be a point between c_{2n} and c_1. We also assume that the arcs in **A** have been relabeled such that $i < j$ implies $h[i] < h[j]$. Two arcs A_i and A_j overlap, denoted by $I_i \times I_j$, if they share a common point. Two arcs A_i and A_j are non-overlapping, denoted by $I_i \div I_j$, if they share no common point. Let $first(P)$ and $last(P)$ denote the first and the last element in path P, respectively. Two paths P_1 and P_2 may be concatenated, denoted by $P_1 \bigcirc P_2$, if $last(P_1) \times first(P_2)$.

Let R be a set of arcs covering $h[1]$ $(= c_1)$. We sort the arcs in R on decreasing order of their head points. That is, if $R = \{r_1, \cdots, r_g\}$, then $h[r_1] > \cdots > h[r_g]$. For each arc r in R, we define $\alpha(r)$ as the part of arc r such that $\alpha(r_i) = [c_0 + \varepsilon_i, t[r_i]]$, and $c_0 + \varepsilon_1 < \cdots < c_0 + \varepsilon_g < c_1$. We denote $\alpha(R)$ to be $\{\alpha(r_1), \cdots, \alpha(r_g)\}$. Let $I = A - R$. I is, in fact, an interval model. Unless mentioned otherwise, in this paper we assume that a set of intervals is sorted in increasing order of their head points.

A path P, denoted $PR(I)$, is called a (longest) *prefix path* in an interval model $I = \{I_1, \cdots, I_k\}$, if it includes a prefix set of maximal number of intervals $I_1, \cdots, I_i$ for some $i \le k$. Note that in a prefix path P, if $i < j$, I_i does not necessarily appear before I_j. But if I_j appears in P, then I_i must also be in P. Find_PR(I, P, S) is the algorithm to find the prefix path from I, where S = I − P. This algorithm takes $O(nlogn)$ time [4].

A (minimal) *partition* of **I** is defined to be $\{M_0, \cdots, M_i\}$ such that (1) there is an HP in each set M_j, for $j = 0, \cdots, i$; For convenience, without misunderstanding let M_j also denote such an HP path; (2) i is the smallest number such that $\{M_0, \cdots, M_i\}$ satisfies (1). Such a number i is referred to as the *partition number* (PN) of **I**, denoted $PN(I)$. $PN(I) + 1$ is, in fact, the minimal number of node disjoint paths. $PN(I)$ can be found by repeatedly applying algorithm Find_PR to **I**. Specifically, let $H = \{H_0, \cdots, H_k\}$ such that $H_0 = PR(I)$ and $H_i = PR(I - H_0 - \cdots - H_{i-1})$, for $i=1, \cdots, k$, where $\sum_{i=0}^{k} H_i = I$. $PN(I)$ is thus k.

The algorithm, Find_P1, uses the so-called first-fit strategy to find a smallest ordered subset R_1 of R such that $PN(\alpha(R_1)+I)$ is minimal. For $R = \{r_1, \cdots, r_g\}$, let R_1 be empty initially. If $PN(\{\alpha(r_1)\}+I)$ is reduced by 1, then r_1 is added to R_1. We continue the process, for $i = 2, \cdots, g$, if $PN(\{\alpha(r_i)\}+R_1+I)$ is smaller than $PN(R_1+I)$, then r_i is added into R_1.

Let $R_2 = R - R_1 = \{t_1, \cdots, t_j\}$, where $h[t_1] > \cdots > h[t_j]$. For each arc t in R_2, we define $\beta(t)$ as the part of arc t such that $\beta(t_i) = [h[t_i], c_{2n} + \varepsilon_i]$ and $\beta(t_i)$ does not cover c_0, where $\varepsilon_1 < \cdots < \varepsilon_j$ for $1 \le i \le j$.

Algorithm Find_HP first finds P_1, then P_2, where P_2 is the prefix path in $S_1 + \beta(R_2)$. P_2 can be found by applying algorithm Find_PR($S_1 + \beta(R_2)$, P_2, S_2). Note that if P_2 is empty, then P_1 is an HP for **A**. Assume that P_2 is not empty. If $A_1 \times last(P_2)$, then $P_2 \bigcirc P_1$ is an HP. Otherwise, we should consider the possibility of finding another special type of subset of R, $R_1^{(1)}$, such that $(P_1 =)\ PR(I + \alpha(R_1)) - \alpha(R_1) \subset (P_1^{(1)} =)\ PR(I + \alpha(R_1^{(1)})) - \alpha(R_1^{(1)})$. Let $P_2^{(1)}$ be obtained by applying algorithm Find_PR($S_1^{(1)} + \beta(R_2^{(1)})$, $P_2^{(1)}$, $S_2^{(1)}$), where $R_2^{(1)} = R - R_1^{(1)}$. If $A_1 \times last(P_2^{(1)})$, $P_2^{(1)} \bigcirc P_1^{(1)}$ is an HP in **A**.

In [4], we proved that R_1 and $R_1^{(1)}$ possess the following desirable properties. (1) R_1 is the smallest ordered subset of R such that $PN(I + \alpha(R)) = PN(I + \alpha(R_1))$; (2) $R_1^{(1)}$ is the smallest ordered subset of R such that $PR(I + \alpha(R_1^{(1)})) - \alpha(R_1^{(1)}) = PR(I + \alpha(R)) - \alpha(R)$. These properties are crucial to prove nonexistence of HP, if neither $A_1 \times last(P_2)$ nor $A_1 \times last(P_2^{(1)})$. The detailed algorithm for finding HP in **A** with the complete correctness proof is given in [4].

References

[1] A. Brandstadt and D. Kratsch, On the restriction of some NP-complete problems to permutation graphs, *Report No N/84/80, Sektion Mathematik der Friedrich-Schiller-Universitat Jena, East Germany, 1986.*

[2] D. Gouyou-Beauchamps, The HC problem is polynomial for 4-connected planar graphs, *SIAM J. Comput.* **11** (3) (1982) 529-539.

[3] J. M. Keil, Finding Hamiltonian circuit in interval graphs, *Inform. Process. Lett.* **20** (1985) 201-206.

[4] Y. D. Liang, C. Rhee, S. K. Dhall, S. Lakshmivarahan, *Finding Hamiltonian paths in circular-arc graphs,* Technical Report, School of Electrical Engineering and Computer Science, Univ. of Oklahoma, 1991

[5] G. K. Manacher, T. A. Mankus and A. J. Smith, An optimum $O(nlogn)$ algorithm for finding a canonical Hamiltonian circuit in a set of intervals, *Inform. Process. Lett.* **35** (1990) 205-211

Shortest Paths in Orthogonal Graphs *

Sandeep Bhatia[1] Daniel S. Hirschberg[2] and Isaac D. Scherson[2]

[1]Department of Electrical Engineering
Princeton University
Princeton, NJ 08544

[2]Department of Information and Computer Science
University of California
Irvine, CA 92717

Abstract

Orthogonal graphs were introduced as a simple but powerful tool for the description and analysis of a class of interconnection networks. Routing, and hence finding shortest paths between any two nodes of an orthogonal graph, becomes an important problem. It is shown in this paper that routing in this class of graphs reduces to a node covering problem in the bipartite coverage graph of the orthogonal graph. A minimum cover clearly leads to a shortest path. In general, the problem of finding the minimum node cover in a bipartite graph is NP-complete. However, the bipartite coverage graphs corresponding to orthogonal graphs have a regular pattern of edges. This allows the development of a routing algorithm which results in a minimum cover. The procedure executes in time polynomial in the number of bit-nodes of the bipartite graph. It therefore results in a shortest path algorithm whose time complexity is quadratic in the logarithm of the number of nodes in the original orthogonal graph.

Keywords: Orthogonal Graphs, Coverage Graphs, Routing, Minimum Cover, Shortest Path.

1 Introduction

Orthogonal graphs were recently introduced as a unified framework for the description and analysis of a large class of interconnection networks [10, 11, 12]. The graphs are defined by a construction rule based on three parameters, namely the number of nodes (2^m), a

---

*This work was supported in part by the Air Force Office of Scientific Research under grant number AFOSR-90-0144.

mask length $n \leq m$, and a set of connectivity modes $Q \subseteq \{0, 1, ..., m-1\}$. Two nodes labeled with m-bit strings are connected if there exists an integer $q \in Q$ such that the node labels differ in at most $m - n$ bits starting at bit position $(q + n) \bmod m$. The resulting structures correspond to spanning bus hypercube-like structures and describe systems such as the binary hypercube, Batcher's Multidimensional Access (MDA) memories, spanning bus hypermeshes, most known multistage interconnection networks, and many others. In this paper we address the routing problem in orthogonal graphs and suggest a simple algorithm to find shortest paths. It is clear that, given that orthogonal graphs describe such a large class of systems, the proposed algorithm applies to those systems as well.

It is shown that the connectivity of an orthogonal graph can be depicted by means of a bipartite graph which consists, on one side, of nodes representing label bits and, on the other side, cover nodes corresponding to integers in the set Q. A path between any two nodes is an ordered sequence of nodes which differ in at most $m - n$ bits covered by some mode defined in Q. The routing problem reduces to that of finding a cover set of a subset of label bits, those in which source and destination labels differ. However, the general problem of finding a minimum cover in arbitrary bipartite graphs is NP-complete. Fortunately, some properties of the coverage graph of orthogonal graphs allow the development of a simple routing procedure which results in a shortest path. We show here that a minimum solution can be found in at most $O(m^2)$ steps, where m is the logarithm of the number of nodes in the orthogonal graph.

In Section 2, a review of orthogonal graphs is given for completeness and to provide the nomenclature used in the paper. Section 3 describes the algorithm to identify a minimum length path between any two nodes of an orthogonal graph.

2 Review of Orthogonal Graphs

An orthogonal graph is an undirected graph $G(n, m, Q)$ which has 2^m nodes labeled distinctly from Y_m, the set of m-bit binary labels. The edges of graph G are defined by a construction rule parameterized by n, which is an integer less than m, and Q, the *set of modes*, which is a subset of the integers $\{0, ..., m-1\}$. Let y_i denote the i-th bit of a label, $\underline{y} \in Y_m$, and let $\oplus$ denote the exclusive-OR operation.

The edge construction rule is as follows: Two *distinct* nodes labeled $\underline{y}$ and $\underline{y}'$ are connected by an edge if and only if there exists a $q \in Q$ such that

$$\sum_{i=q}^{(q+n) \bmod m} y_i \oplus y'_i = 0$$

The nodes are said to be *orthogonal mode q*. This is denoted symbolically as $\underline{y} \perp_q \underline{y}'$.

The following properties are ascribed to orthogonal graphs:

1. Connectivity: An orthogonal graph $G(n, m, Q)$ is connected if and only if

   $$\forall i \in [0, m-1], \exists q \in Q \textit{ such that } |(q+n) \bmod m - i| < (m-n)$$

   We say that mode q *covers* bit position i.

2. Disjoint modes: Q is said to be a *disjoint set of modes* if and only if for all $i \in [0, m-1]$ there is a unique mode $q \in Q$ which covers bit position i.

3. Omega graphs: A connected orthogonal graph with a disjoint set of modes is called an *omega graph.* We denote these graphs as $\omega G(n, m)$ because, in order to satisfy connectivity and disjointness, there must be some integer ω such that $m = \omega(m-n)$ and $Q = \{0, (m-n), 2(m-n), ..., (\omega-1)(m-n)\}$. Note that any omega graph with $Q = \{l, l+(m-n), ..., l+(\omega-1)(m-n)\}$ for any integer $0 < l < m-n$ is isomorphic to the graph for $l = 0$. The diameter of an omega graph is the size of Q ($\#Q$).

3 Routing

The routing problem in orthogonal graphs was outlined in [11, 12]. Because of the similarity between this problem and that of Boolean minimization, it was conjectured that finding a shortest path would require an exhaustive search as a solution is not unique. In this paper we use some properties of the bipartite coverage graph of an orthogonal graph to develop an algorithm which results in a shortest path solution.

As was outlined in [11, 12], to find a path between nodes $\underline{y}$ and $\underline{y}'$, we first compute a routing tag $\underline{t} = \underline{y} \oplus \underline{y}'$. The tag $\underline{t}$ has 1's in all those bit positions in which the nodes labels differ.

We define the *coverage graph* for an orthogonal graph $G(n, m, Q)$ to be a bipartite graph G* in which the left-hand-side nodes (bit nodes) correspond to label bits and the right-hand-side nodes correspond to modes in Q (q-modes or cover nodes). A q-mode is adjacent to a bit node if and only if the mode covers that bit position. A connected orthogonal graph will have a coverage graph in which all bit nodes are connected to at least one cover node. For the remainder of our discussion we consider only connected orthogonal graphs.

A set α of cover nodes in the coverage graph corresponds to (generates) a set of paths in the orthogonal graph. If $q \in \alpha$, then a node labeled with a bit string $\underline{y}$ is adjacent to a set of nodes whose labels differ, from $\underline{y}$, in at most $m - n$ bits starting at bit position $(q + n) \bmod m$. Path sequences can be generated as follows. Starting from a given source node in the orthogonal graph, iterate obtaining the label of the next node in a path by inverting the bits of the current node label at bit positions corresponding to a subset of the bit nodes adjacent to one of the cover nodes of α. Note that a regular $(m - n)$-ary tree can be constructed for each node in the orthogonal graph. This tree represents all possible shortest paths available from node $\underline{y}$ to all other nodes.

We define the *routing graph* as the subgraph of the coverage graph obtained by deleting the bit nodes corresponding to the 0-bits in $\underline{t}$ and the edges associated with those deleted nodes. The routing problem is reduced to finding a minimum subset of the q-modes that cover the bit nodes in the routing graph.

In Figure 1, the left bipartite graph is the general converage graph for $G(5, 8, \{0, 2, 3, 5, 6\})$. An example routing problem is given in the right graph of Figure 1. The routing tag, $\underline{t} = \underline{y} \oplus \underline{y}'$, is shown at the top and the circled nodes, corresponding to 1-bit positions in $\underline{t}$, are those that must be covered. A solution minimum covering set of q-modes corresponds to a *routing mask* set which need not be applied in any predefined order. Note that the minimum covering set is not necessarily unique. For this example, the three solution routing mask sets are indicated.

It is worth noting here that if $G(n, m, Q)$ is an ω-graph, the coverage set will contain all cover nodes in the routing graph as bit nodes are uniquely covered. Such a case is trivial and will not be discussed any further.

We define the following notation :

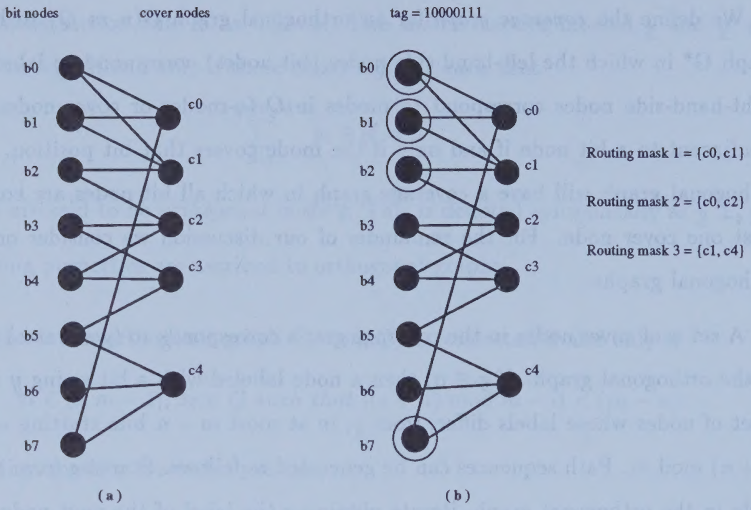

bit nodes
cover nodes
b0
b1
b2
b3
b4
b5
b6
b7
c0
c1
c2
c3
c4
(a)
tag = 10000111
b0
b1
b2
b3
b4
b5
b6
b7
c0
c1
c2
c3
c4
Routing mask 1 = {c0, c1}
Routing mask 2 = {c0, c2}
Routing mask 3 = {c1, c4}
(b)

Figure 1: (a) Coverage graph for $G(5,8,\{0,2,3,5,6\})$ (b) Routing problem for path from 01101011 to 1110100

b_i : a bit node in the coverage graph.

c_j : a cover node in the coverage graph.

$adj(\alpha)$: the set of nodes adjacent to node α in the coverage graph. See Figure 2 for examples.

β_i : the set of bit nodes in the routing graph that are adjacent to the i^{th} cover node of the minimum cover. Note that the union of all β_i's gives all 1-bits in the routing tag $\underline{t}$.

Some of the β_i's may have non-empty intersection. Arbitrarily, remove common bit nodes from intersecting β_i's so that they become a disjoint partition of the set of 1-bits in $\underline{t}$.

$\hat{\beta}_i$: a subset of bit nodes in the routing graph that are adjacent to the i^{th} cover node of the minimum cover so that the set of $\hat{\beta}_i$'s is a disjoint partition of the 1-bits in $\underline{t}$.

The set of $\hat{\beta}_i$'s gives the minimum path (indices of the intermediate nodes, if any) for the routing tag $\underline{t}$. Starting from the source node, the index of the next node in a minimum path is obtained by inverting the index at bit positions corresponding to one of the $\hat{\beta}_i$'s. Each of the $\hat{\beta}_i$'s is considered only once. The destination node is reached when all the $\hat{\beta}_i$'s have been considered.

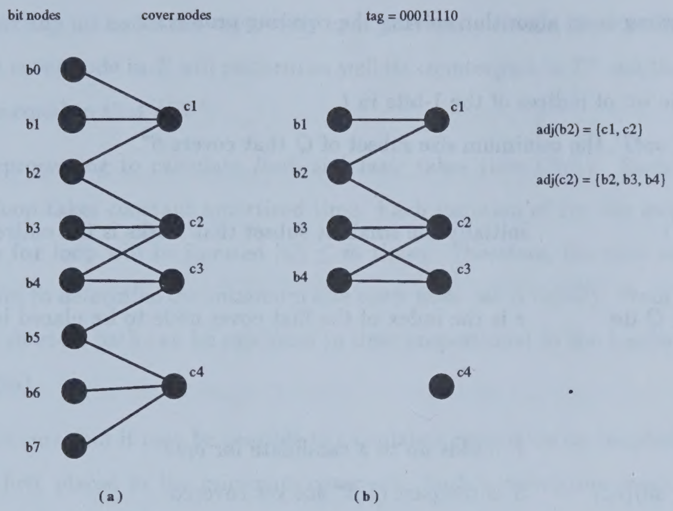


Figure 2: A typical coverage graph (a) Coverage graph for $G(5, 8, \{0, 3, 5, 6\})$, (b) Routing graph for path from 01000111 to 01011001

For the example of Figure 2, c_1 and c_2 form a minimum cover. The bit nodes adjacent to them are $\beta_1 = \{b_1, b_2\}$ and $\beta_2 = \{b_2, b_3, b_4\}$. Let $\hat{\beta}_1 = \{b_1, b_2\}$ and $\hat{\beta}_2 = \{b_3, b_4\}$. Therefore, a minimum-length path from 01000111 to 01011001 is

$$01000111 - 01000001 - 01011001$$

The order of using the $\hat{\beta}_i$'s is not important, so another minimum path is

$$01000111 - 01011111 - 01011001$$

For each cover node c_j in the coverage graph, *adj*(c_j) is a contiguous subset of the bit nodes if they are considered as forming a ring $b_0, b_1, ..., b_{m-1}, b_0$. Accordingly, for cover node c_j, we denote *adj*(c_j) by the pair of indices *firstb*(c_j) and *lastb*(c_j). If *firstb*$(c_j) \leq$ *lastb*(c_j) then $adj(c_j) = \{b_{firstb(c_j)}, ..., b_{lastb(c_j)}\}$. Otherwise, $adj(c_j) = \{b_{firstb(c_j)}, ..., b_{m-1}, b_0, ..., b_{lastb(c_j)}\}$. It follows that, in the routing graph, *adj*(c_j) is also a contiguous subset of the bit nodes.

Similarly, for each bit node b_i in the coverage graph, *adj*(b_i) is a contiguous subset of the cover nodes if they are considered as forming a ring $c_0, c_1, ..., c_{n-1}, c_0$. Accordingly, for bit node b_i, we denote *adj*(b_i) by the pair of indices *firstc*(b_i) and *lastc*(b_i). If *firstc*$(b_i) \leq$ *lastc*(b_i) then $adj(b_i) = \{c_{firstc(b_i)}, ..., c_{lastc(b_i)}\}$. Otherwise, $adj(b_i) = \{c_{firstc(b_i)}, ..., c_{n-1}, c_0, ..., c_{lastc(b_i)}\}$. In the routing graph, *adj*(b_i) is also a contiguous subset of the cover nodes.

The following is an algorithm solving the covering problem :

Let β^* be the set of indices of the 1-bits in $\underline{t}$.
We will find *optT*, the minimum size subset of Q that covers β^*.

```
$optT \leftarrow Q$                       initially the smallest subset that works is the entire set

foreach $z \in Q$ do                      $z$ is the index of the first cover node to be placed in $T$

  $x \leftarrow z$
  $T \leftarrow \{c_x\}$                    $T$ builds up to a candidate for $optT$
  $S \leftarrow \beta^* - adj(c_x)$           $S$ is the part of $\beta^*$ not yet covered

  while $S$ is non-empty do
    if $\exists i \in S$ s.t. $i > lastb(c_x)$
    then $nextb \leftarrow \min\{i \in S | i > lastb(c_x)\}$
    else $nextb \leftarrow \min\{i \in S\}$
                                          $nextb$ is the (circular) first index $i$ s.t. $b_i$ not yet covered
    $x \leftarrow lastc(b_{nextb})$           $x$ is the (circular) last index $j$ s.t. $c_j$ covers $b_{nextb}$
    $T \leftarrow T \cup \{c_x\}$              add one more cover node to $T$
    $S \leftarrow S - adj(c_x)$               remove the bit nodes in $S$ covered by $c_x$
  endWhile

  if $(|T| < |optT|)$ then $optT \leftarrow T$

endFor
```

The correctness of the algorithm is easily shown. One of the iterations of the **for** loop will initialize the set T to contain a cover node of a minimum cover set T^*. As we next see, this iteration will produce a minimum cover node set T. Each iteration of the **while** loop determines the circular first bit node b_{nextb} not yet covered by T and then covers it with the circular last possible cover node c_x. Any other cover node that could cover b_{nextb}

will not cover any bit nodes not covered by c_x or previously chosen cover nodes. Therefore each chosen cover node in T will perform as well its counterpart in T^* and the cardinality of T will be equal to that of T^*.

The preprocessing to calculate *lastb* and *lastc* takes time $O(m)$. Each iteration of the **while** loop takes constant amortized time. Each iteration of the **for** loop takes time $O(m)$. The **for** loop will be iterated $|Q| \leq m$ times. Therefore, the time complexity of the algorithm to determine the minimum size cover node set is $O(m^2)$. From the solution cover set, a shortest path can be produced in time proportional to the length of the path, which is $O(m)$.

We conjecture that it may be possible to calculate a restriction on the choices of which item to be first placed in the minimum cover set. Such a restriction would reduce the complexity of this algorithm correspondingly.

4 Conclusions

In this paper, an algorithm to identify a shortest path in an orthogonal graph is presented. Since orthogonal graphs can be used to describe various interconnection networks, this provides a tool to find a shortest path from any node to any other node in a general interconnection network.

The work can be further expanded to identify several different shortest paths, if more than one shortest path exists, or to find a minimum length path in the presence of a fault in a set of nodes and links in the interconnection network.

References

[1] K. E. Batcher, *The Multidimensional Access Memory in STARAN,* IEEE Transactions on Computers, Vol. C-26, No. 2, February 1977, pp. 172-177.

[2] R. E. Buehrer et al., *The ETH Multiprocessor EMPRESS: A Dynamically Reconfigurable MIMD System*, IEEE Transactions on Computers, Vol. C-31, No. 11, November 1982, pp. 1035-1044.

[3] L. N. Bhuyan and D. P. Agrawal, *Generalized Hypercube and Hyperbus Structures for a Computer Network,* IEEE Transactions on Computers, Vol. C-33, No. 4, April 1984, pp. 323-333.

[4] R. P. Grimaldi, *Discrete and Combinatorial Mathematics. An Applied Introduction,* Reading, Addison-Wesley Publishing Co., Inc., 1989.

[5] F. Harary, *Graph Theory,* Reading, Addison-Wesley Publishing Co., Inc., 1969.

[6] K. Hwang and D. Kim, *Generalization of Orthogonal Multiprocessor for Massively Parallel Computation,* Proceedings of Frontiers 88, 2nd Symposium on the Frontiers of Massively Parallel Computation.

[7] K. Hwang, P. S. Tseng and D. Kim, *An Orthogonal Multiprocessor for Large-Grain Scientific Computations,* IEEE Transactions on Computers, Vol. C-38, No. 1, January 1989, pp. 47-61.

[8] I. D. Scherson and Y. Ma, *Vector Computations in an Orthogonal Memory Access Multiprocessing System,* Proceedings of the 8th Symposium on Computer Arithmetic, May 1987, pp. 28-37. February 1989, pp.238-249.

[9] I. D. Scherson and Y. Ma, *Analysis and Applications of the Orthogonal Access Multiprocessor,* The Journal of Parallel and Distributed Computing, Vol. 7, No. 2, October 1989, pp.232-255.

[10] I. D. Scherson, *Definition and Analysis of a Class of Spanning Bus Orthogonal Multiprocessing Systems,* Proceedings of the 1990 ACM Computer Science Conference, February 19-22, 1990, Washington DC, pp. 194-200.

[11] I. D. Scherson, *Orthogonal Graphs and the Analysis and Construction of a Class of Multistage Interconnection Networks,* Proceedings of the 1990 International Conference on Parallel Processing, August 1990.

[12] I. D. Scherson, *Orthogonal Graphs for the Construction of a Class of Interconnection Networks,* IEEE Transactions on Parallel and Distributed Systems, Vol. 2, No. 1, January 1991, pp. 3-19.

[13] I. D. Scherson, Ashish Mehra and Jennifer Rexford, *Toward Scalable Algorithms for Multidimensional Access Systems* Proceedings of the IEEE 1990 Symposium on the Frontiers of Massively Parallel Computation, Maryland, October 1990.

[14] C. L. Seitz, *The Cosmic Cube,* Communications of the ACM, Vol. 28, No. 1, January 1985, pp. 22-33.

[15] H. J. Siegel, *Interconnection Networks for Large Scale Parallel Processing : Theory and Case Studies,* Reading, Lexington Books, Lexington, Mass. 1984.

[16] H. S. Stone, *High Performance Computer Architecture,* Reading, MA, Addison Wesley, 1987.

[17] A. Tucker, *Applied Combinatorics,* Reading, John Wiley & Sons, Inc., 1980.

Bounded 2 contraction of graphs *

Badrinath Ramamurthy and Mukkai S Krishnamoorthy
Department of Computer Science
Rensselaer Polytechnic Institute
Troy, NY 12180

Abstract

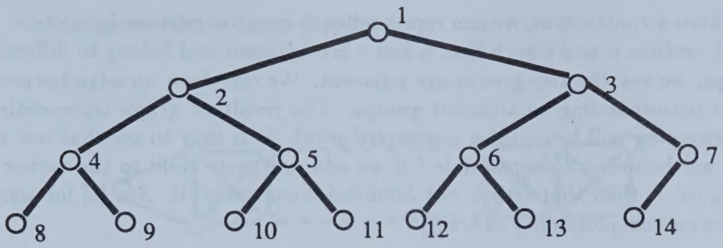
In this paper we are concerned with some of the issues relating to the problem of embedding program graphs into a network architecture of the kind introduced in [1]. In [1] the authors show how to embed certain regular graphs into that architecture. We are interested in detecting graphs with certain contractability properties. In particular, we present a $O(|V|^3)$ algorithm for 2 - contractable graphs. We also enumerate properties of such graphs. This problem is also related to the embedding problem in VLSI [3] [4]. Further the results of this problem could be used in prettier drawing of graphs and trees.

1 Introduction

In this paper we are concerned with some of the issues relating to the problem of embedding program graphs into a network architecture of the kind introduced in [1]. In that report the authors refer to the OPAM architecture. A program consisting of many processes can be represented as a process graph, with the processes forming the vertices and the edges representing communication between two processes. When this graph is mapped into the OPAM architecture, it is necessary to partition the vertices into groups so that a certain constraint is satisfied on the number of links the group of vertices

---

*Supported in part by the NSF under grant CDA-8805910.



Partitions are: (1,11),(2,5),(4,10),(8,9),(3,14),(6,7) and (12,13).

Figure 1: A tree and its partitoning

can have with other vertices. In [1] the authors show how to embed certain regular graphs into that architecture. We are interested in detecting graphs with certain contractability properties. This problem can be viewed as a partitioning problem, with each partition consists of atmost two vertices. The embedding problem is also related to the embedding problem in VLSI [3] [4]. Further the results of this problem could be used in prettier drawing of graphs and trees.

We use standard graph theory terminology as in [2]. We now formally define our contraction scheme.

Definitions:

1. A *p-contraction* of a graph G is a grouping of vertices of G into non empty groups, each of size no larger than p.

2. A *bounded p-contraction* of G is a p-contraction of G in which for each group of vertices, the number of distinct groups that its vertices are adjacent to is no larger than the number of vertices in that group.

For any value of p, if G is connected and has a bounded p-contraction, then maximum degree of a vertex in G is no more than $p^2 + p - 1$.

2 Bounded 2 contraction

We now study the bounded 2-contraction of graphs. As an example, see Figure 1, where a tree and its bounded 2 contraction are given.

Objective: We attempt to solve the question- 'Given a graph G, does their exist a bounded 2-contraction for G ?'

Given a contraction, we can represent each group of vertices by a node. If G has vertices u and v such that u and v are adjacent and belong to different groups, we say the two groups are adjacent. We can draw an edge between nodes corresponding to adjacent groups. The resultant graph representing this grouping will be called a *contracted graph.* It is easy to see that not all trees are bounded 2-contractible (if we add one more child to the vertex 7 in Figure 1, then that tree is not bounded 2-contractable. See [1] for some results on contractability of trees)

2.1 Properties of bounded 2-contractable graphs

For all our purposes *contracted graphs* means bounded 2-contracted graphs. Since each node in a contracted graph is adjacent to at most two other nodes, the contracted graph is either a cycle or a path.

Definition: A graph G has a *rooted contraction* at a vertex u, if there exists a contraction of G into a path such that u is in one of the two end vertices in an end node of the path.

Here is a list of properties which are easy to verify.

Property 1 . If G has odd size, then any contracted graph of G, if one exists, is a path. Further, exactly one of the end nodes of the path contains exactly one vertex of G and that vertex has degree 1 or 2.

Property 2 . If a contraction of G is a path, then the vertices of G in the end nodes of the path have degree no more than 3.

Property 3 . Every contractable graph is a subgraph of some maximal contractable graph on the same number of vertices. Contractable graphs are neither subgraph nor supergraph closed. The maximal contractable graphs look like those in figures 2 and 3. Here one may view two vertices on the same vertical to be paired in a node of the contracted graph. One may also visualize these graphs as two rows of vertices being paired with eachother. Such visualization helps understand some of the properties of these graphs.

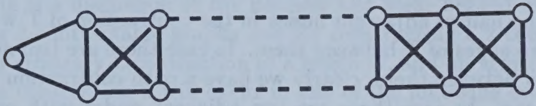


Figure 2: A maximal odd graph

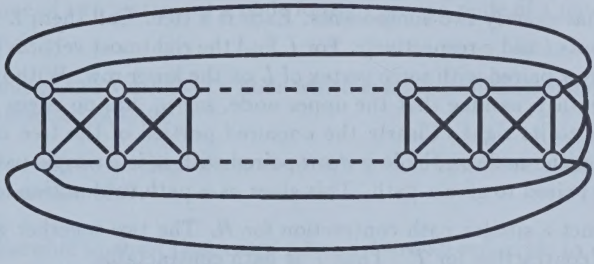


Figure 3: A maximal even graph

Property 4 . If G is contractable then the graph formed by removing any edge is contractable. (contractability is closed under edge removal).

Property 5 . If a graph G that has a contraction which is a path, then G is planar . The converse is not true. Further there are contractable planar graphs which are not contractable to paths.

Property 6 . If a graph G has a contraction, then the maximum number of leaf vertices that G can have is $\lfloor n/2 \rfloor + 2$.

Property 7 . If T is a tree and T is contractable, then there is a contraction of T as a path.

Proof:

Case 1: $|V(T)|$ is odd : Proof is trivial from property 1.

Case 2: $|V(T)|$ is even : Let T be the spanning tree of the maximal 2 contractable graph G on $|V(T)|$ vertices. Since T is a tree, there exists at least one pair of adjacent nodes in the contraction of T which have one or zero edges of T between them. In case there are two nodes with zero edges between them, clearly we have a path contraction of T, and we are done. In case there are two adjacent nodes with one edge e between them, we can embed T in a maximal graph G as in Figure 3, so that that edge e appears among the edges between the leftmost and the rightmost pairs of vertices.

$T - e$ has exactly two components. Each is a tree. Call them L and R with roots l and r respectively. For L find the rightmost vertices on the upper row paired with some vertex of L on the lower row. Without loss of generality, assume that the upper node, say u_l, has no edges to the vertices to its right. Clearly the unpaired portion of the tree coming out from the node on the lower row paired with u_l is a simple path and can be paired to give a path. This gives us a path contraction for L.

Construct a similar path contraction for R. The two together give us a path contraction for T . Thus T is path contractable.

Thus every tree that is contractable is contractable to a path.

It is easy to see that any contraction derived in the above manner can have atmost one node more in the contracted graph than a minimum node contraction.

Property 8 . Assume a tree T has a rooted contraction at a vertex u. Let v be a vertex of degree 3 or more such that the distance between u and v is maximum. Then every vertex of degree 3 in T is either on this path or is adjacent to a vertex on this path and every vertex of degree more than 3 is on this path. We call this path a **spine** at v.

3 Algorithms to detect bounded 2 contractability

The following is a discussion of the method and ideas behind our algorithm for detecting 2-contractable graphs and for constructing a 2-contraction.

Given an arbitrary 2-contraction of a graph G, and an arbitrary vertex u of G, in that contraction u will occur in one of the following four ways:

1. as the single element of a node forming an end node of a path contraction of G,
2. as one of two vertices of a node forming an end node of a path contraction of G,
3. as one of two vertices of a node which is an intermidiate node of a path contraction of G, or
4. as one of two vertices of a node which is an intermidiate node of a cycle contraction of G.

It is possible to show that cases 1,3 and 4 are all reducible to case 2. So we concentrate on building an algorithm for case 2, i.e we attempt to answer the question: *Given a graph G and a vertex u of G, is there a contraction of G with u in an end node of the contraction ?* Our algorithms answer this question and build a contraction for the graph.

We procede now to outline algorithms for the case where G is a biconnected graph, and the case where G is a tree.

3.1 The algorithm for biconnected graphs

Give a graph G and a vertex v, the following algorithm finds a path contraction of G at v if one exists. At any time in the algorithm, if a certain grouping forces three vertices to be grouped, we call that a *failure*.

Algorithm **TestPath***:*

1. The possible candidates to group v with are at distance 1 or 2 from v, so try each possibility. Since it is biconnected, it can be shown that any grouping that does not lead to a *failure* will work as well as any

other. Once all forced pairings are done, we may either be done, or end up with a single vertex.

2. If done then quit; else consider the new vertex to be v, and go back to step 1, repeating it with the rest of the graph.

If in step 1, we don't find any vertex to match v with without failure, then the contraction we are looking for doesn't exist. Clearly, this algorithm takes $O(|V|)$ time. But cases 3 and 4, discussed earlier, require us to see if the particular vertex v can be paired with any other vertex. Hence we may have to repeat the above algorithm about $|V|$ times, hence the total time to construct a contraction is $O(|V|^2)$.

3.2 The algorithm for trees

The problem with applying the algorithm for biconnected graphs to trees is because in step 1 of the algorithm, we can never look for vertices at a constant distance from v, and further it is *not* true that *any* pairing that does not force a *failure* is equally good. This implies a backtracking may be required and hence the algorithm becomes slow.

We avoid this using knowledge of *Property 8* that we described in Section 2. The intuition behind this algorithm is as follows. Each time we get a "single" node at the end of a forced pairing, we need to consider that, in addition to the nodes that are at distance 1 or 2, leaf nodes could also possibly pair up with the node. Further, we want to make that choice which will leave maximum room for the rest of the tree to be paired. Refering to *Property 8*, we want to leave as much room as possible for the degree 3 and above nodes along the spine at v, which have not been paired yet. In implementing the algorithm one can be a little more efficient and not try to pair a vertex with all leaf vertices.

We can think of the given vertex u as a kind of root, since we will always expect this vertex to be in one end node of the contraction. Hence we call this algorithm the *FindRootedContraction* algorithm.

Recall that we are given a vertex u and we assume that the tree will be of type 2. Consider the part of a tree shown in Figure 4. The spine is shown in thicker lines. Assume we want to do *FindRootedContraction* at u.

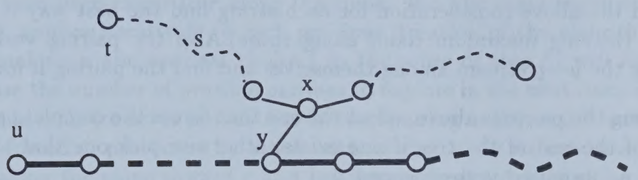


Figure 4: Trying a rooted contraction at u

Property 8 implies that all vertices not on the spine are paired up with spine vertices (except possibly for some vertices at the very end). So we explore the possibility of pairing strings of vertices like the x—t string (labelled s) with vertices along the stream of vertices between u and y. However, we can only pair t with a vertex at an even distance from u, because once we pair t with some vertex w, we are implying that all the vertices from u till w (not including w) will be paired among themselves, and this requires an even number of vertices.

It may be possible that t can be paired with more than one vertex along the u—y stream without forcing a failure. Also, more than one string may be incident at x or y and we may have more than one vertex such as x to considered. If more than one of these seem feasible, we are faced with the question of which one to choose. At this point our guiding principle is to chose the pairing that will either force the rest of the tree giving a successful result or else leave maximum room for the rest of the tree to be paired along the spine. In this sense our algorithm is a greedy algorithm.

We define a *string* at a vertex v to be a path starting from v and terminating in a degree 1 vertex such that all intermediate vertices have degree 2.

Algorithm **FindRootedContraction**:

1. Travel from u along the path out of u to the nearest degree 3 or more vertex. Call it y.

2. Find candidate strings s_i and their tips t_i.

3. For each candidate string determine if its tip should be pulled inward to match a vertex nearest to u or pushed outward to pair with a vertex farthest from u, along the u—y path.

4. With the above consideration for each string find the best way it can pair (leaving maximum room along spine).Also try pairing vertices along the u—y stream among themselves and find the pairing it forces.

5. Among the pairings above, select the one that forces the complete pairing of the rest of the tree if one exists, otherwise pick one that stops with an unpaired vertex nearest u.

6. If the complete tree has been paired then quit with a success, otherwise go back to step 1., with the current vertex as u and repeat.

In case in step 5 all strings produce *failures*, then the contraction we are looking for does not exist.

In step 2, there are only finitely many vertices and strings at these vertices that can be candidate strings. But checking if a certain candidate pairs appropriartely can take $O(|V|)$ time. Hence the running time of the above algorithm is $O(|V|^2)$. But, as before, u may belong to an intermediate node (cases 3 and 4). Hence the running time of our algorithm for detecting bounded 2 contractable trees is $O(|V|^3)$.

In fact, this algorithm motivates the idea of an algorithm that works for detecting contractability of graphs in general.

It is clear that when the algorithm above halts with a success, we definitely have a rooted contraction of the tree. We'll now sketch a proof that when the algorithm halts at step 5 with *failure*, then there is no rooted contraction at the root, say r, that we started at. Failure at step 5 indicates that none of the candidate strings could be paired up with the u—y stream of vertices, and that the attempt to pair vertices along the stream among themselves also leads to forcing a *failure*. Assume now that there is a rooted contraction at r. This implies that either a. some string (not a candidate string at y) further down the spine from y is to pair up with vertices along the spine but before y, or b. some candidate string at y is to be paired up with some vertices along the spine between r and u in addition to the vertices on the u—y stream. Case a can not be true, as it can easily be seen that attempting such a piring leads to *failure*. Next assume case b is true. So the vertex, currently

u, should appear nearer to r along the spine. Since step 5 indicates we have made the best possible choice at this juncture, we could have only erred in the steps before, i.e the vertex u' that we started the previous iteration at must have been closer to r along the spine. But the same argument, about step 5, applies iteratively to each previous iteration of the algorithm. Note that making a non optimal choice (in the sense of step 5) will never help increase the number of possible pairings to explore in the next iteration. Also an optimal choice never prevents or forces a pairing in the next iteration, that is not forced or prevented by a non optimal choice. So we can never move u back along the spine toward r and hence case b cannot hold either. Thus if our algorithm fails, there is no rooted contraction at r.

4 Conclusion and future work

Other open questions that we intend to look into include the following. Is detecting bounded p-contractability NP-Hard ? Is bounded 2-contractability efficiently parallelisable? It will also be interesting to find heuristic algorithms for k-contractibility. Also, the complexity of k-contractibility for trees and planar graphs seems like an interesting topic.

References

[1] A. Barak and E. Schenfeld, *Embedding Classical Communication Topologies in the OPAM Architecture*, Technical Report 90-12, Department of Computer Science, Hebrew University of Jerusalem, Israel.

[2] J. A. Bondy and U. S. R. Murty, *Graph Theory With Applications*, North-Holland, 1976.

[3] N. Deo, M. S. Krishnamoorthy and M. Langston, *Exact and Approximate Solutions for the Gate Matrix Layout Problem*, IEEE Trans. Computer Aided Design, 6, 1987, pp. 79-84.

[4] R. H. Möhring, *Graph Problems Related to Gate Matrix Layout and PLA Folding*, Computational Graph Theory, edited by G. Tinhofer, E. Mayr, H. Noltemeier and M. Syslo, Springer-Verlag, New York, 1990, pp. 17-51.

A Parallel Algorithm for Finding Minimum Cutsets in Reducible Graphs

Eliezer Dekel and Jie Hu
Department of Computer Science
The University of Texas at Dallas
e-mail: {dekel, jhu}@utdallas.edu

Abstract: We present a parallel algorithm for finding minimum cutsets in reducible graphs. For a reducible graph that has N nodes our algorithm runs in $O(\log^3 N)$ time using $O(N^3/\log N)$ PE's on the EREW P-RAM model of computation.

Keywords: reducible graph, cutset, complexity, EREW P-RAM.

1. Introduction

Processes, in particular computer programs, are often represented by directed graphs. The analysis and manipulation of system modeled as graphs require a selection of minimal subset of the nodes that cuts all the cycles in the graphs. This set is referred to as a minimal cutset (or feedback node set). Two common application areas that use cutsets are program verification and code optimization [HU1],[F]. Reducing the size of the node set, that cuts all the cycles, usually leads to a simpler and more efficient analysis.

The problem of finding a minimal cutset in a directed graph is NP-complete [K]. A. Shamir [S] showed that for an important class of directed graphs, reducible graphs [HU1], there is a serial linear time algorithm for finding a minimum cutset. Most directed graphs produced from flow charts of computer programs, (or flow graphs), are reducible. We refer the reader to the paper by Shamir [S] for more background for the problem. The ongoing research in the area of vectorizing and parallelizing compilers brought more interest in reducible graphs. Recognizing loops and optimizing execution around them are of key importance in this area. Recently Lucas and Sackrowitz showed that testing whether a given directed graph with N nodes is reducible can be done in $O(\log^2 N)$ time using $O(N^3/\log N)$ PE's [LS].

In this paper we present a parallel algorithm for finding a minimum cutset in a reducible graph in $O(\log^3 N)$ time using $O(N^3/\log N)$ PE's on EREW P-RAM. Finding an efficient parallel algorithm for this problem is particularly interesting because the sequential algorithm employs a Depth First Search tree [S]. Depth First Search is believed to be inherently sequential [R]. Our parallel algorithm utilizes the tree pruning technique developed by Dekel and Hu [DH2]. We refer the reader to the excellent review by Karp and Ramachandran [KR] for discussion on the different parallel models. In section 2 we present the problem and some definitions. In section 3 we present a parallel algorithm for finding the minimum cutset in a special type of reducible graph. The main result is presented in section 4. Detailed proofs and examples can be found in the paper by Dekel and Hu [DH1].

2. Definitions

For the sake of brevity we review here only those definitions that are necessary to understand our algorithms. See the paper by Hecht and Ullman [HU2], the paper by Shamir [S], and the paper by Dekel and Hu [DH1] for more definitions and detailed discussions of reducible graphs and their properties.

We use $G(V,E)$ to denote a directed graph. A cutset of G is a subset of the nodes $S \subseteq V$ such that any cycle in graph G contains at least one node from S. A *minimum cutset* of a given graph is a cutset of minimum size. The *DAG* of a directed graph G is a maximal acyclic graph that is a subgraph of G. A reducible graph has unique DAG. Every cycle of G contains exact a non DAG edge. Let (v,u) be a non DAG edge, then u is called a *head* and v is its *tail*.

The sequential algorithm [S] is based on the following principle: If a node u is a head and there is a path from u to one of its tails such that every node (except u itself) on the path has been determined not to be in the cutset, then u is determined to be in the cutset. We later refer to this principle as the *decision principle*. It implies that an algorithm in which every node in cutset is selected according to the decision principle gives a minimum cutset. The complexity of this serial algorithm is $O(|V|+|E|)$.

3. Minimum cutset in a branch

In developing our parallel algorithm we first handle a special type of reducible graphs. We name this type of reducible graph a 'branch'. A reducible graph K(V,E) is a *branch* if the DAG of K is a chain.

The following are equivalent for a branch.

(1) K is a reducible graph and it has unique DAG that is a chain.

(2) The edges of K can be partitioned into chain edges and back edges.

(3) Each back edge creates a cycle in K.

In a branch K, let U_1 and U_2 denote the node sets of cycles C_1 and C_2 respectively. If U_1 is contained in U_2 then cutpoint of C_1 is also a cutpoint of C_2. Let E_1 denote the set of edges of the DAG of K (the edges of the chain) and E_2 denote the set of backedges. The nodes of K are labeled according to the postorder of the DAG of K. Applying the decision principle to the postorder labeled DAG of K we observe the following property: let (u,v) and (x,y) be two backedges of K, if $x \leq u$ and $y \geq v$ then backedge (x,y) is redundant for the purpose of finding a minimal cutset in K and can be ignored. After deleting all the redundant back edges from a branch K we get a graph K' that we call a *simple branch*. A simple branch K' with n nodes has at most $(N\text{-}1)$ back edges. Each node can be a head at most once and be a tail at most once. It is easy to see that a minimal cutset of a simple branch is also a minimal cutset of the corresponding branch.

Let C_u denote the cycle of a simple branch K' that is formed by back edge (v,u).

Let t_u denote the tail of head u in simple branch K'.

Lemma 3.1: If a head x is in cycle C_u and $x<v<u$ where v is also a head, then: (a) x is in cycle C_v, and (b) v is in cycle C_u.

Procedure Serial_Branch_MinCut (D)

```
{Input is a head-dependency graph $D$}
$S_1 \leftarrow \{v \mid v$ has no incoming edge$\}$;
$S_2 \leftarrow \varnothing$;
repeat
    if a node $u$ has an immediate predecessor that is in $S_1$, then put $u$ in $S_2$.
    if a node $u$ has all its immediate predecessors in $S_2$ then put $u$ in $S_1$.
until every node is either in $S_1$ or in $S_2$.
$S_1$ is a cutset.
```

Algorithm 3.1

We now construct D, the *head–dependency* graph of a simple branch K'. The nodes of a head-dependency graph are the heads of the simple branch and (u,v) is an edge in D iff u is in cycle C_v. We proceed to show how to find a cutset of a simple branch (hence a branch) using the head-dependency graph of a simple branch (Algorithm 3.1).

Procedure Serial_Branch_MinCut uses two sets to partition the nodes of the head-dependency graph D. In steps 1 and 2 of the procedure we initialize these sets. Clearly all nodes of head-dependency graph D are candidates for membership in a minimal cutset. Set S_1 is initialized with nodes that have to be in the cutset. These are the nodes that have no incoming edges in D. Set S_2 has nodes that are determined not to be in the cut set as members and it is initially set to be empty. The loop in lines 3 to 6 is executed until all nodes are determined either not in the cutset or as members of the minimal cutset. The correctness of the algorithm is proved in the following lemmas.

Lemma 3.2: In every iteration of the loop, (lines 3-6, Algorithm 3.1), there is at most one immediate predecessor in S_1 for any node.

Lemma 3.3: A node u is placed in S_1 by procedure Serial_Branch_MinCut, (Algorithm 3.1), iff there is a path in K' from u to t_u such that every node on that path (except u) is either in S_2 or is non-head.

Lemma 3.4: S_1 is a minimum cutset of simple branch K'.

Corollary 3.1: S_1 is a minimum cutset of K.

In our discussion we assume that the information for the branch includes the partition of its edges to DAG edges and back edges. Although this information is not typically available as input, it is available in our context as we shall show later. To reduce a branch to a simple branch takes O(M) time where M is the number of edges in the branch. The construction of the head-dependency graph takes O(m) time where m is the number of edges in the head-dependency graph. Procedure Serial_Branch_MinCut, (Algorithm 3.1) takes O(m) time, since every edge is checked by some node once. Thus we can find a minimum cutset of a branch in linear time.

```
Procedure P_BRANCH_CUTSET (K);
{ Input is a branch K. Nodes are labeled in postorder of its DAG. Edges are partitioned into
DAG edges and back edges. }
A.  Simple_Branch;   {Produce simple branch K' by deleting redundant back edges.}
B.  Head_Graph;      {Construct head-dependency graph from K'.}
C.  {Find a minimum cutset for branch K}
1   Cutset of K←∅;
2   for each block of the head-dependency graph in parallel do
3           Block_Cutset;  {Determine the minimum cutset for each block.}
4   Cutset of K← Cutset of K∪ cutset of block;
end;{P_BRANCH_CUTSET}
```

Algorithm 3.2

A head-dependency graph may contain several weakly connected components. We call such a component a *block*. Procedure P_BRANCH_CUTSET is a parallel algorithm for computing a minimum cutset for a given branch K, (Algorithm 3.2). It is a parallel implementation of the ideas in Procedure Serial_Branch_MinCut, (Algorithm 3.1), and the discussion leads to it.

```
procedure Simple_Branch;
1.  if a node $u$ is a head then in parallel do
        $v \leftarrow \max\{\text{tails of } u\}$;   {find the closest tail}
    Delete the backedges $(w,u)$ if $w \neq v$;
2.  if a node $v$ is a tail then in parallel do
        $u \leftarrow \min\{\text{heads of } v\}$;   {find the closest head}
    Delete backedge $(v,w)$ if $w \neq u$;
3.  Sort backedges in increasing order according to the tails;
    for each backedge $(u,v)$ in parallel do
    if the next backedge $(x,y)$ has $x > u$ and $y < v$ then delete $(u,v)$;
end; {Simple_Branch}
```

Algorithm 3.3

```
procedure Head_Graph;
{Input is a simple branch $K'$. Output is a set of head-dependency graphs.}
1)  {initialize}
    Mark all heads;
    Eliminate all non-head nodes from consideration;
    Initiate a head-dependency graph $G^*=(N^*,E^*)$ where
        $N^* \leftarrow \{\text{all heads}\}$;
        $E^* \leftarrow \emptyset$;
2)  {build the head dependency graph}
    for each head w in parallel do
        for each backedge $(u,v)$ in parallel do
            if $v > w \geq u$ then $E^* \leftarrow E^* \cup \{(w,v)\}$;
3)  {identify blocks}
    Every node in $G^*$ without incoming edges is the beginning of a block;
    Every node in $G^*$ without outgoing edges is the end of a block;
    for each beginning of a block u in parallel do {Construct a block}
        block_end($u$) $\leftarrow \min\{v \mid v \text{ is an end of block and } v \geq u\}$;
        nodes of block $\leftarrow \{v \mid u \leq v \leq \text{block\_end}(u) \text{ and } v \text{ is a head}\}$;
        edges of block $\leftarrow \{(x,y) \mid x,y \in E^*\}$;
end.{Head_Graph}
```

Algorithm 3.4

There are three main steps in Procedure P_BRANCH_CUTSET, (Algorithm 3.2). In the subsequent paragraphs we discuss the parallel implementation of each of these steps.

Procedure Simple_Branch (Algorithm 3.3) has 3 steps. The implementation of these steps is quite straight forward. It can perform its job in O(log$N$) time using $N^2/\log N$ PE's. Using an adjacency matrix to represent the graph, we can find the tail with the minimal label for each node in O(log$N$) time with O(N/log$N$) PE's [H] [DNS]. Deletion of the redundant edges found in step 1 or step 2 can be done in O(log$N$) time using N/log$N$ PE's, for each head. Therefore steps 1 and 2 of the procedure can be performed in O(log$N$) time using N^2/log$N$ PE's for the whole graph. There are at most N-1 edges left in step 3 of the procedure, sorting can be performed in O(log$N$) time with N^2/log$N$ PE's [C]. The edge deletion can be done in constant time.

```
procedure Block_Cutset;
{Input is a head dependency graph $D(N_D, E_D)$. Output is the cutset.}
1.  Relabel the nodes according to their ranks in $D$;
2.  for every node u in parallel do min_adjacent($u$)$\leftarrow$min{$v$|($v$,$u$)$\in E_D$};
3.  for every node u in parallel do construct $g_u(N_u,E_u)$ where
    $N_u \leftarrow \{y | y \in N_D$ and $y \geq u\}$;
    $E_u \leftarrow \emptyset$;
    for each node $y > u$ in parallel do {each node has one in-edge}
            if min_adjacent($y$)$\leq u$ then $E_u = E_u \cup (u,y)$
              else $E_u \leftarrow E_u \cup$(min_adjacent($y$),$y$);
4.  for all nodes $i$ and for all $g_u$ in parallel do
        S($i$,$u$)$\leftarrow$ 'undetermined';  { S($i$,$u$) denotes the status of node $i$ in $g_u$ }
    for all $g_u$ in parallel do
            S($u$,$u$)$\leftarrow$'in-cutset';
            max-in($u$)$\leftarrow u$;
5.  for i=1 to $\lceil \log N_D \rceil$ do
            for each $g_u$ in parallel do
                    5.1 for all nodes $x \in N_u$ do
                    cp($x$,$u$)$\leftarrow$0; { zero the number of copies requested }
                    5.2 for each edge ($x$,$y$) in parallel do
                    {directly identify nodes that are not in cutset}
                      if ($u \leq x < u+2^{i-1}$) and (($u+2^{i-1}$)$\leq y < u+2^i$) then
                            5.2.1 if S($x$,$u$)='in-cutset' then S($y$,$u$)$\leftarrow$'out';
                            5.2.2 if S($x$,$u$)='out' and $x<$max-in($u$) then S($y$,$u$)$\leftarrow$'out';
                    5.3 $w$=min{$x$| S($x$,$u$)='undetermined' and $x \leq u+2^i$};
                    5.4 if there is such $w$ then do
                            5.4.1 cp($u$,$w$)$\leftarrow$1;
                            5.4.2 {to avoid conflicts in 5.4.3 }
                              make $\sum_{x \in N_u}$ cp($x$,$u$) copies of S(*,$u$);
                            5.4.3 {merge in values from $g_w$}
                        for each $x \in N_u$ in parallel do
                                if $w \leq x < u+2^i$ then S($x$,$u$)$\leftarrow$S($x$,$w$);
                    5.5 max-in($u$)$\leftarrow$max{$y$| S($y$,$u$)=in-cutset};
6.  Cutset of the block$\leftarrow${$u$| S($u$,1)=in-cutset};
end.{Block_Cutset}
```

Algorithm 3.5

It is easy to see that procedure Head_Graph, (Algorithm 3.4), can be performed in O(log$N$) time using O(N^2/log$N$) PE's on the EREW P-RAM model of computation. The initialization step can be done in constant time using O(N) PE's. In step 2, a backedge may produce O(N) edges in the head-dependency graph. Since the input is a simple branch there are no write conflicts. This step can be performed in O(1) time using O(N^2) PE's or in O(log$N$)

time using O(N^2/log$N$) PE's. In step 3, finding the minimum can be done in O(log$N$) time using o(N^2/log$N$) PE's for all nodes that are the beginning of blocks. Block identification can be done in O(1) using O(N^2) PE's or O(log$N$) using O(N^2/log$N$) PE's.

In the first step of procedure Block_Cutset, (Algorithm 3.5), we relabel the nodes for each block (The root of the block is determined in step 3 of procedure Head_Graph, (Algorithm 3.4)). $g_u(N_u,E_u)$ is the subgraph rooted at u. We will see these g_u's allow us to maximize the parallel computation without conflicts. Each node is connected from the min_adjacent node computed in step 2. Adjustments are made for cases where min_adjacent is smaller than the root u. S(i,u), the status of node i in subgraph g_u, is initially set to 'undetermined' by step 4. In this step we also initialize S(u,u) to 'in_cutset' and set the max_in variable associated with each root node u to u.

The loop in step 5 is the most significant part of the algorithm. Its correctness is not apparent and we will discuss it in Lemma 3.5. In iteration i of the loop we work on nodes y in the range $u+2^{i-1} \leq y < u+2^i$. We identify the nodes in the cutset in two fashions. One is direct, that is, we determine the nodes that are not in the cutset, (S(y,u)='out'), by looking at their immediate predecessors, step 5.2. The second way in which we determine membership in the cutset is by merging values of nodes calculated in other subtrees g_w where $w>u$. This allows us to break the sequential nature of "look at the predecessor" idea.

As for the complexity of procedure Block_Cutset (Algorithm 3.2): The relabeling can be performed in O(log$N$) using O(N/log$N$) PE's [KR]. If a PE is assigned to each edge in step 2, there will be no read conflict. Computing the minimum can be accomplished in O(log$N$) using O(N^2/log$N$) PE's. In Step 3 we need n copies of the head dependency graphs, in order to avoid read conflict. The replication and the construction of the g_u graphs can be performed in O(log$N$) time with O(N^2/log$N$) PE's. Step 4 can be performed in O(1) using O(N^2) PE's or O(log$N$) time using O(N^2/log$N$) PE's. Finally, Step 5 requires O(log N) iterations. In each iteration we zero out vector cp(*,u) ,(statement 5.1). Variable cp is used in order to ascertain the number of copies of g_u that are needed to avoid read conflicts in statement 5.4.3. Clearly each of the statements, (5.1-5.5), that make up the loop can be performed in O(log$N$) time with O(N^2/log$N$) PE's. Observe that there are at most N copies to be made in statement 5.4.2. By Lemma 3.2 there is at most one immediate predecessor that could be in-cutset, there are no read conflicts in statement 5.2. Therefore step 5 can be performed in O(log^2N) time using O(N^2/log$N$) PE's. In conclusion, procedure Block_Cutset can be performed in O(log^2N) time with O(N^2/log$N$) PE's on EREW P-RAM model of computation.

Define D_u to be a subgraph of the head-dependency graph D that contains only nodes with label $\geq u$ and their incident edges. Using D_u, rather then g_u in procedure Block_Cutset, variables min_adjacent and max-in are no longer required. We do need to use though, O(N^3/log$N$) PE's, since each D_u may have N^2 edges. It is easy to observe that D_u and g_u are equivalent for the purpose of finding the minimal cutset of a brunch. We discuss the correctness of procedure Block_Cutset in the following paragraphs. To facilitate the discussion we use D_u and g_u interchangeably.

Lemma 3.5: After the i^{th} iteration of the loop in step 5 of procedure Block_Cutset, S(y,u) for $u \leq y < u+2^i$ is set to 'in-cutset' iff y is put in S_1 by procedure Serial_Branch_MinCut, (Algorithm 3.1), with input D_u.

Corollary 3.2: After log$N$ iterations, S(y, 1)=in-cutset iff y is in S_1 by procedure Serial_Branch_MinCut, (Algorithm 3.1) on head-dependency graph D.

Theorem 3.1: Procedure P_BRANCH_CUTSET finds a minimum cutset for a branch.

We can conclude that Procedure P_BRANCH__CUTSET (Algorithm 3.2) can be performed in O(log^2N) time using O(N^2/log$N$) PE's or in O(log$N$) time using O(N^3/log$N$) PE's on the EREW P-RAM model of computation. This can be easily determined from the complexity analysis of the procedures .

4. Parallel algorithm for minimum cutset of a reducible graph

Let *T* be a directed tree with root *r*. For every leaf *v* of *T*, there is a node *w* that is a predecessor of *v* such that every node between *w* and *v* has exactly one incoming edge and one outgoing edge, i. e., it is a chain from *w* to *v* in *T*. We call a chain in *T* that ends with a leaf an active chain. If we prune all the active chains from the tree *T*, new leaves and new active chains will be created. This pruning operation can continue until the root r becomes part of an active chain. Dekel and Hu show that there could be at most O(log*N*) pruning iterations for a rooted tree with *N* nodes, [DH2]. This pruning process is special case of the compress operation in the tree contraction method, [DNP] [MR]. The difference is that in each iteration only the chains that contain leaves are compressed. We refer the readers to the paper by Dekel and Hu [DH2] for more details of pruning decomposition. If T is a spanning tree of a reducible graph, then the subgraph related to the active chain is a branch. Our algorithm finds a minimum cutset of a reducible graph by successively identifying branches, and finding the cutset of those branches. The procedure P_Cutset, Algorithm 4.1, is the skeleton of the algorithm.

```
procedure P_Cutset:
{Input is a reducible graph $G$.}
1.  Construct a breadth first spanning tree of reducible graph $G$;
2.  repeat
3.  Identify all the branches;
4.  Prune the branches;
5.  Determine the cutset of each branch;
6.  until the root is determined.
```

Algorithm 4.1

Before we present the details of the algorithm we need to establish the correctness of our approach. We know that a reducible graph G has unique backedge set on any Depth First Search (DFS) tree of *G* [S]. We first establish that the same is true for a Breadth First Search (BFS) tree of *G*.

Lemma 4.1: In a reducible graph *G*, an edge (u,v) is a backedge of a DFS spanning tree of *G* iff it is a backedge of any BFS spanning tree of *G*.

Corollary 4.1: Any BFS spanning tree of a reducible graph partitions the edge set into backedges and DAG edges and the DAG edge set can be partitioned into tree edges and cross edges.

Corollary 4.2: An algorithm that is based on the decision principle can find a minimum cutset of a reducible graph using the DAG defined by a BFS tree of the graph.

Consider the first iteration of the repeat loop of procedure P_Cutset, (algorithm 4.1). Initially all nodes of the reducible graph *G* are undetermined. Using the active chains of the BFS tree *T* of *G*, we construct active branches of *G*. Node *v* is in an active branch of *G* iff *v* is in an active chain of *T* and edge (u,v) is in an active branch iff both *u* and *v* are nodes in the branch.

Lemma 4.2: In the first iteration of the repeat loop, node u is determined to be in the cutset of an active branch by our algorithm iff u is determined to be in cutset of the reducible graph G by the decision principle.

Corollary 4.3: The nodes that are determined to be in the cutset by procedure P_BRANCH_CUTSET in the first iteration are in the minimum cutset of the reducible graph determined by the decision principle.

Some of the successors in G of the active branches in the i^{th} iteration are already processed. Several of those successors are determined to be in the minimum cutset and others are determined not to be in the cutset. The assumptions that allowed us to process the active branches in the first iteration independently do not hold. We now present a transformation that allows us to treat active branches at any iteration independently.

```
procedure BRANCH_NORMALIZING;
{Input $B$, a branch that some of its successors have been determined. Edges between two
nodes of the branch are either tree edges or back edges.}
  1) Construct a graph $g' \leftarrow (N_{g'}, E_{g'})$ such that
        $N_{g'} \leftarrow \{v \mid v$ is in $B$ or $v$ is a node that is determined not to be in the cutset$\}$;
        $E_{g'} \leftarrow \{(u,v) \mid u,v$ are in $g'$ but are not both in $B\}$;
  2) Compute the transitive closure on $g'$;
  3) {Construct normalized branch.}
    3.1 Construct a graph $h \leftarrow (N_h, E_h)$, initially
          $N_h \leftarrow \{y \mid y$ is undetermined$\}$;
          $E_h \leftarrow \emptyset$;
    3.2 for each pair of nodes $x,y$ in $B$ in parallel do
          if $x < y$ and $x$ can reach $y$ in $g'$ then
            $E_h \leftarrow E_h \cup \{(x,y)\}$;
    3.3 $B^N \leftarrow h \cup \{$edges of $B\}$;
end.{BRANCH_NORMALIZING}
```

Algorithm 4.2

Procedure BRANCH_NORMALIZING (Algorithm 4.2) processes an active branch B with some predetermined successors and produces a normalized branch with no successors. Once we obtained the normalized branch we can apply our branch cutset algorithm. In the first step of the procedure we construct a graph g' that includes the nodes of branch B and all nodes that are already determined not to be in the cutset. In step 2 of the algorithm we compute the transitive closure of g' to find the exact relationships between branch nodes and successor nodes.

Based on the computation of step 2 we construct in step 3 our normalized branch. In 3.1 we initialize the graph to contain the nodes of B, (undetermined nodes). The set of edges is initially empty. We next consider the relation between all pairs of nodes in the transitive closure of g' in step 3.2. Edge (x,y) is added to our normalized branch if $x < y$ and there is a path from x to y in g'.

Lemma 4.3: In the normalized branch B^N, head u can reach its tail v iff there is a path from u to v in the reducible graph G and every node on the path (except u and v) is not included in the cutset by the serial algorithm.

The complexity of procedure BRANCH_NORMALIZING is dominated by the transitive closure step. This can be done in O($\log^2 N$) time using $N^3/\log N$ PE's on the EREW P-RAM model of computation [DNS] [H]. The input is represented as an NxN matrix where N is the number of nodes involved in normalizing process. It is easy to see that all the other steps can be computed in O($\log N$) time with O($N^2/\log N$) PE's. Thus BRANCH_NORMALIZING can be wrapped up in O($\log^2 N$) time with O($N^3/\log N$) PE's.

```
Procedure MIN_CUTSET_REDUCIBLE_G;
        {Input is a reducible graph $G$}
1.      Construct a BFS tree $T$ of $G$;
2.      Label the nodes of $T$ in postorder;
3.      for each node $u \in T$ do
                DECISION#($u$)←outdegree($u$);
                every node is undetermined initially;
4.      repeat
                4.1 Identify all active branches;
                        {find consecutive nodes with DECISION#=1 starting bottom up
                        from each node with DECISION#=0}
                4.2 for every active branch $B$ in parallel do
                        4.2.1 BRANCH_NORMALIZING;
                        4.2.2 P_BRANCH_CUTSET;
                4.3 if a node $u$ is newly determined and $v$ is its parent in $T$ then
                        DECISION#($v$)←DECISION#($v$)-1;
                4.4 if a node $y$ is determined in-cutset then delete $y$ from $G$;
        until root is determined;
5.      Minimum cutset←{$y$|$y$ is determined 'in-cutset'};
end. {MIN_CUTSET_REDUCIBLE_G}
```

Algorithm 4.3

After obtaining B^N we can use P_BRANCH_CUTSET to find a minimum cutset of the branch B^N. Notice that procedure BRANCH_NORMALIZING can produce a branch where some cycles contain more than one backedges. It wont cost problem since P_BRANCH_CUTSET will delete the redundent backedges.

We are now ready to present the parallel algorithm for finding a minimum cutset of a reducible graph G (Algorithm 4.3):

In Algorithm 4.3, variable DECISION# in MIN_CUTSET_REDUCIBLE_G is used to identify the active branches. It is part of the data structure of the tree and is updated after each iteration for the remaining undetermined nodes. If u has DECISION#(u)=0, u will be a leaf in next iteration, if u has DECISION#(u)=1 then u is (or will be) in some active branch.

Lemma 4.4: Procedure MIN_CUTSET_REDUCIBLE_G finds a minimum cutset of reducible graph G.

It is easy to see that algorithm MIN_CUTSET_REDUCIBLE_G can be done in O($\log^3 N$) time by using O($N^3/\log N$) PE's on EREW P-RAM model of computation. Step 1 can be performed in O($\log^2 N$) using O($N^3/\log N$) PE's,[DNS] [GM]. Step 2 can be done in O($\log N$) with O(N) PE's using the Euler tour technique[TV]. The computations of steps 3 4.1 and 4.3 can be performed in O($\log N$) by O($N^2/\log N$) PE's using standard techniques. As we discussed earlier Step 4.2.1 and 4.2.2 can be done n O($\log^2 N$) with O($N^2/\log N$) PE's. Finally Step 4.4 can be done in O(1) with O(N^2) PE's. The number of iterations is bounded by O(log$N$) [DH2].

Theorem4.1: Algorithm MIN_CUTSET_REDUCIBLE_G finds a minimum cutset of a reducible graph in O($\log^3 N$) time by using O($N^3/\log N$) processors on the EREW P-RAM model of computation.

Reference:

[C] R. Cole, "Parallel Merge Sort," *Proc.27th FOCS*, pp.511-516, 1986.

[DH1] E. Dekel and J. Hu, "A Parallel Algorithm for Finding Minimum Cutsets in Reducible Graphs," *UTD Tech. Rep. UTDCS-6-91*, 1991.

[DH2] E. Dekel and J. Hu, "Efficient Parallel Graph Decomposition and Biconnected Components in Graphs on EREW," *UTD Tech. Rep. UTDCS-7-91*, 1991.

[DNS] E. Dekel, D. Nassimi and S. Sahni, "Parallel matrix and graph algorithms," *SIAM J. Comput.* vol 10, No 4, pp.657-675, 1981.

[DNP] E. Dekel, S. Ntafos and S. T. Peng, "The compression trees and their applications," *Proc. Int. Conf. Parallel Processing*, pp.132-139, 1987.

[F] R. W. Floyd, "Assigning meaning to programs," *Proc. Symp. Appl. Math.*, 19 , pp.19-32, 1967.

[HU1] M. S. Hecht and J. D. Ullman, "Flow graph reducibility," *SIAM J. Comput.* vol 1, No 2, pp.188-202, 1972.

[HU2] M. S. Hecht and J. D. Ullman, "Characterizations of reducible flow graphs," *J. Assoc. Comput. Mech.*, 21, pp.367-375, 1974.

[H] D. S. Hirschberg, "Parallel algorithms for the transitive closure and the connected components problems," *ACM symposium on the Theory of computing*, pp.55-57, 1976.

[GM] H. Gazit and G. L. Miller, "An improved parallel algorithm that computes the BFS numbering of a directed graph," *Inf. Proc. Letters* 28, pp.61-65, 1988.

[K] R. M. Karp, "Reducibility among combinatorial problems," Complexity of Computer Computations, R. E. Miller and J. W. Thatcher, eds., Plenum Press, New York, pp.146-160, 1972.

[KR] R. M. Karp and V. Ramachandran, "A Survey of Parallel Algorithms for Shared-Memory Machines;" Handbook of Theoretical Computer Science, Cambridge, MIT Press, 1990.

[LS] J. M. Lucas and M. G. Sackrowitz, "Efficient parallel algorithms for path problems in directed graphs," *Proc. SPAA*, pp.369-378, 1989.

[MR] G. Miller and J. Reif, "Parallel tree contraction and its application," *Proc. of 26th FOCS*, pp.478-489, 1985.

[P] F. P. Preparata, "New Parallel Sorting Scheme," *IEEE Transactions on Computers*, Vol.C-27, No.7, pp.669-673, 1978.

[S] A. Shamir, "A linear time algorithm for finding minimum cutset in reducible graphs," *SIAM J. Computing* 8, pp.645-655, 1979.

[TV] R. E. Tarjan and U. Vishkin, "An efficient parallel biconnectivity algorithm," *SIAM J. Comput.* vol 14, No 4, pp.862-874, 1984.

Measuring Presortedness in Ensembles of Data Sequences

R.S. Valiveti and B.J. Oommen *
School Of Computer Science
Carleton University
Ottawa, Canada, K1S 5B6

1 Introduction

Sorting is a problem of paramount importance in the field of computer science. The optimal sequential time complexity of sorting a sequence of M elements is $O(M \log M)$. While many optimal techniques are known, there also exist algorithms (e.g. Quicksort) which are not optimal in the worst case sense. These algorithms can be presented with specific sequence of data elements, for which a running time of $O(M^2)$ results. This phenomenon has led researchers to seek alternate sorting methods which never encounter this worst case quadratic behaviour. The key idea motivating this research is to find methods whose time complexity increases "smoothly" as a function of some measure of "presortedness" of the input sequence [3].

The class of "adaptive" sorting methods achieve exactly this goal. These methods monitor some measure of the input sequence, such as the number of inversions [1] or the number of oscillations [2]. Such a measure in essence defines the "distance" of the input data sequence, from its sorted form. Algorithms based on these measures have a time complexity which is $O(M)$ when the "distance" measure is small and approaches the lower bound for sorting as the measure increases [2].

While such adaptive algorithms perform rather well on specific input sequences, an objective comparison of these algorithms with other existing methods (or even among themselves) cannot be achieved by just studying the "distance metrics" of individual sequences. An example will help clarify this point. Let us suppose that the set of data elements to be sorted is $\{1, 2, 3, 4, 5\}$. If an Insertion Sort (IS) is presented with the sequence $\pi_1 = \langle 1, 2, 3, 5, 4\rangle$ the sorting is achieved by performing exactly one swap and $O(M)$ comparisons. However if the input sequence was $\pi_2 = \langle 5, 4, 3, 2, 1\rangle$, $O(M^2)$ swaps would be necessary.

As opposed to the Insertion Sort, let us suppose that the same data sequences were presented to an adaptive scheme, such as the Adaptive HeapSort (AHS) described in [2]. The AHS would take $O(M)$ operations to sort each of the sequences π_1 and π_2. The question now before us is to objectively compare IS and AHS.

If the data to be sorted was only of the type π_1, clearly the IS is superior. IS only make $O(M)$ comparisons and also avoids the overhead encountered in maintaining and updating the heap used by the AHS. However, if the data set was always of the type π_2, clearly the AHS is superior. In order to objectively compare the two algorithms, not

---

*Both authors partially supported by Natural Sciences and Engineering Research Council Of Canada. The work of the first author was also supported by a Graduate Award from Bell-Northern Research.

only should the presortedness metric of a sequence be considered, but the comparison must also involve the probability distribution on the set of permutations on the data set.

Thus our stand in this paper is that whereas distance metrics such as Inversions [1], Runs, and especially Oscillations [2] are extremely powerful in quantifying the presortedness of a **single** data sequence, they are incapable of characterizing the presortedness of **ensembles** of sequences. Instead of assigning metrics to each data sequence contained in an ensemble, we assign a single Ensemble Presortedness measure to the ensemble as a whole. This single parameter reflects the divergence of the various sequences in this ensemble from the purely sorted from. This is the fundamental contribution of this paper.

Strictly speaking, given a set of M elements, the probability associated with each possible data sequence can be fully defined by enumerating $M!$ probabilities. Apart from this characterization being unrealistic and impractical, it does not provide a single parameter with which we can compare two ensembles of data sequences. In this light, as alluded to above, we shall present a new metric called the Ensemble Presortedness metric. This metric is completely specified by a single parameter ρ. Given the parameter ρ, the Distribution on all possible data sequences is completely described by means of an $M -$ dimensional probability vector. The i^{th} component of this vector is recursively defined in terms of the $(i-1)^{st}$, and is obtained by merely multiplying the latter by ρ. Indeed if $\rho = 0$, it represents an ensemble consisting only of the sorted sequence. Furthermore we shall show that as ρ increases, the number of inversions increases.

At this point it is appropriate to compare our metric with other metrics that could be envisioned. A simple example of such an ensemble presortedness metric is the average number of inversions in the data sequences found in the ensemble. The main characteristic of these measures of presortedness is that they quantify the extent of presortedness in an ensemble, but unfortunately they are not capable of *utilizing* the measure generated. Specifically, these metrics cannot be easily integrated into methods for mimicing the statistical characteristics of the ensemble in question, by generating random permutations of data. As opposed to these metrics, our measure forms the basis for a random permutation generation model. As a natural consequence, the measure utilized can be used to generate data sequences which are statistically similar to the ensemble examined. We believe that this concept is new both in the areas of data generation and data analysis.

We shall first show how to generate ensembles of data sequences, given the parameter ρ. Subsequently we also study the problem of evaluating the parameter ρ for a given ensemble. In other words, given an ensemble of data sequences, we present an estimation procedure by which the Maximum Likelihood Estimate (MLE) of the parameter ρ can be obtained. A natural consequence is that we can now compare sorting algorithms not only in terms of their worst case time complexity but also their expected time complexity.

Section 2, presents our model of data sequence generation. This model is based on a fairly general model for permutation generation, called the S-model. Several properties of the ρ-model of data sequence generation are derived in the same section. Section 3 describes the procedures used to obtain the MLE of ρ. Section 4 presents our experimental results.

2 The S-model for Data Sequence Generation

Let $\mathcal{R}$ be the set of records $\{R_1, R_2, \ldots, R_M\}$, where there is an implicit ordering among the elements of $\mathcal{R}$ that $R_i < R_j$ if $i < j$. It is clear that we can extract $M!$ features using a simple estimation procedure on the permutations themselves. Such a strategy is infeasible even when M is as small as 10. We therefore model permutation generators in a fashion which requires M (as opposed to $M!$) parameters. This model, called the S-model, was initially proposed by Oommen and Ng in [4].

In an S-model, we assume that the underlying data generation strategy can be completely specified by a control vector of dimension M. This vector $\mathcal{S}$ is an $M \times 1$ probability vector denoted by $[s_1, s_2, \ldots, s_M]^T$ which satisfies:

$$\sum_{i=1}^{M} s_i = 1.$$

Let $\mathcal{V}$ be any subset of $\mathcal{R}$. We define the vector $\mathcal{S}_\mathcal{V}$ as the **conditional** control vector of (normalized probabilities) in which only the quantities corresponding to the members of $\mathcal{V}$ have non-zero values. Thus $\mathcal{S}_\mathcal{V}$ consists of normalized probabilities $[s_i']$, where:

$$s_i' = \begin{cases} 0 & \text{if} R_i \notin \mathcal{V} \\ \dfrac{s_i}{\sum_{R_i \in \mathcal{V}} s_i} & \text{otherwise} \end{cases} \tag{1}$$

Observe that the vector $\mathcal{S}$ is exactly equivalent to $\mathcal{S}_\mathcal{R}$.

Given the vector parameter $\mathcal{S}$, we assume that the underlying generation strategy is as follows. Let Π be the set of all permutations of the elements of $\mathcal{R}$. For the sake of explanation, let $X = \langle x_1, x_2, \ldots, x_M \rangle$ be a randomly generated data sequence ($X \in \Pi$). The generation of a random data sequence X clearly consists of randomly assigning a position to each element of $\mathcal{R}$. The S-model proposes that this is done by computing the prefix subsequence of X in succession. Initially x_1 is randomly assigned an element in $\mathcal{R}$, based on the control vector $\mathcal{S}$(which is the same as $\mathcal{S}_\mathcal{R}$). By this we mean that element R_i is chosen with probability s_i. The subsequent problem is simply one of generating a (random) sequence of the $(M-1)$ elements of $\mathcal{V} = \mathcal{R} - \{x_1\}$. The S-model proposes that this generation is done using the conditional probability vector $\mathcal{S}_\mathcal{V}$ and the process is recursively repeated by successively updating $\mathcal{V}$. The algorithmic form of this procedure (referred to as Algorithm S-model) is omitted here for the sake of brevity[5].

Having defined the S-model for data sequence generation, the next question is that of specifying the individual components of the vector $\mathcal{S}$. To enable us to have a single parameter distribution metric to compare the presortedness of ensembles, we define s_i as follows:

$$s_i = k\rho^{i-1}$$

Observe that this implies that the complete vector $\mathcal{S}$ is given by:

$$\mathcal{S} = k[1, \rho, \rho^2, \ldots, \rho^{M-1}]^T. \tag{2}$$

where k is a normalizing factor chosen so as to render $\mathcal{S}$ to be a probability vector.

It will be seen that if $\rho < 1$, the components of $\mathcal{S}$ in (2) are in the descending order of magnitude. This fact can be used to show that the data sequence $\langle R_1, R_2, \ldots, R_M \rangle$ occurs with the highest probability. In the limit as $\rho \rightarrow 0$, the ensemble generated by the model consists only of the sorted list. The converse is true if $\rho > 1$, and in the limit as $\rho \to \infty$, the only sequence presented to the sorting algorithm is the reverse of the sorted list. Note that $\rho = 1$ corresponds to the uniformity assumption traditionally made in expected case analysis.

2.1 Properties of the S-model

We now present some of the theoretical properties of the data sequences generated by the S-model.

Theorem 1 *Let $\mathcal{S}$, the (M-dimensional) parameter vector of the S-model be given by (2). Also let $R_u, R_{v_1}, \ldots, R_{v_K}$ be distinct elements of $\mathcal{R}$. Then the probability that the element R_u precedes each one of the elements of the set $\{R_{v_1}, \ldots, R_{v_K}\}$ in any data sequence generated by Algorithm S-model, is given by:*

$$Pr(R_i \prec \{R_{v_1}, \ldots, R_{v_K}\} \mid \mathcal{S}) = \frac{\rho^{u-1}}{\rho^{u-1} + \sum_{j=1}^{K} \rho^{v_j - 1}} \tag{3}$$

Proof:

The proof is quite involved and can be found in [5]. □

Theorem 2 *Let $\mathcal{S} = [1, \rho, \rho^2, \ldots, \rho^{M-1}]$ be the (M-dimensional) control vector of the S-model. Then, in the ensemble of data sequences generated, the expected number of inversions is given by:*

$$E[\text{Inversions}|\rho] = \sum_{j>i} \frac{\rho^{j-i}}{1 + \rho^{j-i}}.$$

Proof:

Let Y_{ij} be an indicator random variable which takes on a value of 1(0) if the record R_i appears before (after) the record R_j in a data sequence. Then the number of inversions in a data sequence X is given by $\sum_{j>i} Y_{ji}$. The required result follows if we note that $E[Y_{ji}] = Pr(R_j \prec R_i)$, which can be written down using Theorem 1. □

It is easy to see that the expected number of inversions is $M(M-1)/2$ when $\rho \to \infty$. Moreover, the expected number of inversions increases with ρ. We are currently investigating the relationship between the parameter ρ and the expected values of various other "presortedness" metrics such as Runs and Oscillations.

3 Estimation of the Presortedness Metric

Till now we have focussed on the problem of obtaining data sequences which are generated based on an M-dimensional vector $\mathcal{S}$ which is specified in terms of a single parameter ρ, called the Ensemble Presortedness metric. In this section, we concentrate on the aspect of estimating the best value of ρ which describes a given data sequence ensemble. We state with a mere sketch of proof, our first result in this regard.

Theorem 3 *The maximum likelihood estimate of the parameter ρ can be obtained only if we maintain $O(2^M)$ statistics.*

Proof:

The proof mainly involves systematically enumerating the counters that appear in the likelihood function which must be maximized to estimate ρ. This result can be proved in the same manner as the analogous result for estimating the parameters of the S-model[6]. □

Theorem 3 has established that in order to rigorously obtain the Maximum Likelihood Estimate for the parameter ρ, we are required to maintain $O(2^M)$ statistics. The amount of information to be extracted from data sequences can be substantially reduced if we only chose to observe events of the form "Record R_u precedes record R_v". This implies the maintenance of $\binom{M}{2}$ counters. Our next result develops the method to be used for estimating the parameters, if only these $O(M^2)$ statistics are available.

Theorem 4 *Let ${}_uN_v$ represent the number of times the element R_u appears before the element R_v, in the ensemble of N data sequences. If only the frequency counts of the form ${}_uN_v$ are available for all pairs u, and v, then the maximum likelihood estimate for the parameter ρ can be obtained as the solution of the non-linear equation given below:*

$$\sum_{i<j} \frac{j-i}{1+\rho^{j-i}} = \sum_{i<j} \frac{{}_iN_j}{N}(j-i). \tag{4}$$

Proof:

The proof is quite involved and can be found in [5]. □

Remark: It must be noted that (4) is a non-linear equation involving ρ. Clearly, an analytic solution is not feasible. In general it can be solved by using a root-solving technique such as the Newton-Raphson (NR) method.

4 Experimental Results

We now examine how effectively the Ensemble Presortedness metric (ρ), reflects the presortedness of various ensembles. To do this, various ensembles of sequences of data were generated and the effect of the mean number of inversions on the MLE of the parameter ρ was studied. In the interest of brevity, we shall only present the results for one set of experiments. The complete set of experimental results can be found in [5].

The method used to generate random data sequences is described in [5]. Using each such sample data sequence, statistics of the form ${}_iN_j$ were updated. This procedure was repeatedly invoked to generate an ensemble of 10,000 samples. After the ensemble was obtained, the MLE of the parameter ρ was obtained by solving (4). Figure 1 shows the variation in ρ for the cases in which M is 16.

Observe that the value of ρ increases monotonically with the increase in the expected number of inversions. When the list is almost sorted, the values of ρ are close to 0. On the other hand, as the expected number of inversions rises to values close to $M*(M-1)/4$, we find that the estimate of ρ approaches unity. This is just as is expected. The use of the metric ρ as a measure of presortedness for an ensemble is obvious.

5 Conclusions

We have proposed a presortedness metric, ρ, for an **ensemble** of data sequences, as opposed to a **single** sequence. This metric captures the divergence of data sequences in the given ensemble, from the sorted list. Various properties of the metric have been stated and a procedure for obtaining its Maximum Likelihood Estimate for a given ensemble has been described. The experimental results from a typical experiment have been presented. Numerous other results are found in the unabridged paper [5].

References

[1] D. E. Knuth, *The Art Of Computer Programming: Vol. 3 Sorting and Searching*, Addison Wesley, Reading, Mass., 1979.

[2] C. Levcopoulos and O. Petersson, "Heapsort — Adapted for Presorted Files", *Proc. of the WADS Workshop, Ottawa, Canada, August 1989*, Springer-Verlag, pp. 499-509.

[3] K. Mehlhorn, "Sorting Presorted Files", *Proc. of the Fourth GI Conf. on Theoretical Computer Science*, pp. 199-212, Springer-Verlag, 1979.

[4] B. J. Oommen and D. T. H. Ng, "Arbitrarily Distributed Random Permutation Generation", *Proceedings of 1989 ACM Comp. Sci. Conf.*, Louisville, KY, Feb 1989, pp. 27-32. Also available as Technical Report SCS-TR-138 from the School Of Computer Science, Carleton University, Ottawa, Canada, K1S 5B6.

[5] R. S. Valiveti and B. J. Oommen, "A Presortedness Metric for Ensembles of Data Sequences", Technical Report SCS-TR-174, School Of Computer Science, Carleton University, Ottawa, Canada, K1S 5B6, May 1990.

[6] R. S. Valiveti and B. J. Oommen, "Recognizing Sources of Random Strings", *IEEE Trans. of Pattern Analysis and Mach. Intell.*, Vol. 13, No. 4, 1991, pp. 386-394.

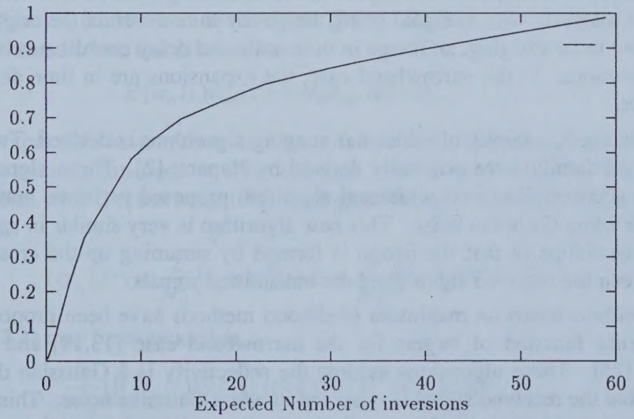


Figure 1: Figure shows the increase in the MLE of ρ as the Expected number of inversions increases. In this case $M = 16$. The maximum number of inversions allowed is 60.

Wideband Radar Imaging with Estimation Theory Applications

Joseph A. O'Sullivan

Electronic Systems and Signals Research Laboratory
Department of Electrical Engineering
Washington University
St. Louis, MO 63130

ABSTRACT

A new imaging algorithm for wideband radar applications based on a transform viewpoint is proposed. The algorithm is shown to give a unitary mapping between the set of received radar data and the desired reflectivity density image. When the received signal is observed in white Gaussian noise, this mapping yields the reflectivity density image plus two-dimensional white Gaussian noise. The relationship to alternative approaches and to selected results from wavelets are discussed.

1. Introduction

A recent paper [1] has proposed an alternative viewpoint for existing radar imaging algorithms and has introduced a new radar imaging algorithm for the wideband radar case. This paper explores this new wideband algorithm and demonstrates limitations of previous wideband algorithms. An extension of the results in [1] to using an analytic representation for the signals is presented.

The background for the wideband algorithm is described in [2,3,4,5,6]. These authors examine the formation of images from wideband radar data based on group representation theory. Similar viewpoints have been taken for narrowband radar imaging [7,4,8,9], where the Heisenberg group is used instead of the affine group. The relationship to wavelet analysis [10,11,12] has been explored in some of these papers and in related papers. In wavelet analysis, an expansion of a time function is sought in terms of two parameters, time scale and time shift, with one goal being the ability to reconstruct the original time function. In wideband radar imaging, an image in time scale and delay coordinates is sought from a set of time functions. In the narrowband case, the expansions are in time delay and frequency coordinates.

In Section 3, a family of wideband imaging algorithms is derived. Two wideband algorithms in the family were originally derived by Naparst [2]. These algorithms do not take noise into account. The new wideband algorithm proposed performs better in the presence of additive white Gaussian noise. This new algorithm is very similar in spirit to narrowband imaging algorithms in that the image is formed by summing up the cross-ambiguity functions between the received signals and the transmitted signals.

Algorithms based on maximum likelihood methods have been proposed for estimating the scattering function of targets for the narrowband case [13,14] and for more general scenarios [15]. These algorithms assume the reflectivity is a Gaussian distributed random field and that the received signal is observed in white Gaussian noise. This model is accurate for the narrowband case [16,17]; its validity for the wideband case is assumed. For additive white Gaussian noise, it is shown below that the scattering function can be recovered from the observed data.

There are other ways to view radar imaging from a transform viewpoint. The most common way is to use tomographic methods as in [18,19,20]. Maass [5] discusses an approximation of the received wideband radar data as integrals over hyperbolae and presents an inversion formula. The approach here is similar in spirit to these approaches in that there are a pair of equations which are inverse transforms of each other. The difference is that the approach here is not based on conventional tomographic techniques.

This paper is oriented to obtaining reconstructions in time scale and time delay coordinates. If the actual desired reconstruction is in physical coordinates (such as range and crossrange), then the mapping from those physical coordinates to the coordinates here must be known. For rigid rotating bodies which do not rotate through a large angle during the illuminations, crossrange is proportional to velocity and range is proportional to delay [21]. In other cases, a direct reconstruction of the reflectivity in physical coordinates may be obtained by considering the appropriate equations for the scenario considered [22,23,24].

In Section 2, the available data is defined. The wideband imaging algorithm is derived in Section 3 and properties of the algorithm are explored in Section 4. Section 5 contains an estimation theoretic approach to imaging based on this method. Proofs of the properties in Sections 4 and 5 are in the appendix.

2. Wideband Radar Data

The received radar signal from the n^{th} transmitted signal equals the sum of the reflected signal and white Gaussian noise

$$r_n(t) = e_n(t) + w_n(t) \ . \tag{2.1}$$

The reflected signal equals the superposition over the target of the transmitted signal time-scaled and time-shifted times the reflectivity density:

$$e_n(t) = \int_0^\infty\int_0^\infty \sqrt{y}\, s_n(y(t-x)) D(x,y)\, dx\, dy \ , \tag{2.2}$$

where $D(x,y)$ is the reflectivity density of the target [1,2]. The time-scale y is related to the velocity in the direction of the transmitter by $y = c+v/(c-v)$, where c is the speed of propagation. The time-shift x is given by $x = 2R_0/(c+v)$, where R_0 is the bulk delay to the target. The integral over x in (2.2) may be extended to $-\infty$ by defining $D(x,y)$ to be 0 for $x<0$. The transmitted signal $s_n(t)$ is the analytic signal [24, p. 7]. $D(x,y)$ is complex valued and $w_n(t)$ is complex valued white Gaussian noise so

$$E[w_n(t) w_m^*(\tau)] = N_0 \delta_{nm} \delta(t-\tau) \ . \tag{2.3}$$

There are two possible assumptions on $D(x,y)$ addressed. The first is that $D(x,y)$ is an unknown but deterministic function. In this case, $e_n(t)$ is also deterministic. Since $w_n(t)$ is white Gaussian noise with spectral height N_0, the loglikelihood function for $r_n(t)$ parameterized by $D(x,y)$ is

$$l(r_n \mid D) = \frac{2}{N_0} \int r_n(t) e_n^*(t)\, dt - \frac{1}{N_0} \int |e_n(t)|^2 dt \ . \tag{2.4}$$

The first term in (2.4) may be expanded as

$$\frac{2}{N_0} \iint \left[\int r_n(t) \sqrt{y}\, s_n^*(y(t-x))\, dt \right] D^*(x,y)\, dx\, dy \ , \tag{2.5}$$

where

$$m_n(x,y) = \int r_n(t) \sqrt{y}\, s_n^*(y(t-x))\, dt \tag{2.6}$$

is the output of a processor matched to the transmitted signal.

If $D(x,y)$ is a random field, the new loglikelihood function is found by taking the logarithm of the expectation of $\exp[l(r|D)]$ using an appropriate measure on D. No matter what this measure is, the only way $r_n(t)$ enters the likelihood function is through $m_n(x,y)$. Thus, for optimal noise performance, the receiver first finds $m_n(x,y)$.

The second possible assumption is a stochastic model for $D(x,y)$. $D(x,y)$ is assumed to be a spatially white zero mean Gaussian random field. So each point is uncorrelated with every other point, and is assumed to have scattering function $\Theta(x,y)$ defined by

$$E[D(x,y)D^*(x',y')] = \Theta(x,y)\delta(x-x')\delta(y-y') . \tag{2.7}$$

This model corresponds to the WSSUS model in Van Trees [16] and the model for diffuse reflections described in [13,17] for narrowband radar. Its validity for wideband radar has not been addressed to our knowledge.

Throughout the remainder of this paper, assumptions about the energy in the transmitted signals $s_n(t)$ are made. The assumption of a given energy in the transmitted signal can be accommodated by either scaling the received signal appropriately or incorporating a scale factor into $D(x,y)$. The value of N_0 must be known for the estimation algorithms to be effective.

3. Wideband Radar Imaging

In this section, the new algorithm is developed, following the approach from [1]. This same approach gives a simplified derivation of the imaging algorithm derived by Naparst [2,3]. This algorithm is based upon a model which ignores the additive noise. The effect of the noise is addressed in Section 5.

The approach is based on a pair of equations, the *analysis* equation and the *synthesis* equation. The synthesis equation takes the form

$$D(x,y) = \sum_n \int e_n(t)\phi_n(t,x,y)dt , \tag{3.1}$$

where the $\phi_n(t,x,y)$ are called the synthesis functions. The $e_n(t)$ are determined by the analysis equation

$$e_n(t) = \iint D(x,y)\psi_n(t,x,y)dxdy . \tag{3.2}$$

One way for these two equations to be consistent is the condition

$$\iint \phi_m(\tau,x,y)\psi_n(t,x,y)dxdy = \delta_{mn}\delta(t-\tau) . \tag{3.3}$$

That is, by using our n^{th} analyzing function on the expansion given by (3.1), the result is the coefficient function for the n^{th} synthesis function.

For our problem, the $e_n(t)$ equal the echoes received from the target when the n^{th} signal is transmitted. Thus, the analysis equation is given by (2.2) and is

$$e_n(t) = \int_0^\infty \int_{-\infty}^\infty \sqrt{y}\, s_n(y(t-x))D(x,y)dxdy . \tag{3.4}$$

Without stating initially our assumptions on the $s_n(t)$, the conditions which define the synthesis functions may be derived. First, substitute the synthesis functions into (3.3), then take the Fourier transform in the t variable to get

$$\int_0^\infty \int_{-\infty}^\infty \phi_m(\tau,x,y)\frac{1}{\sqrt{y}}S_n(f/y)e^{-j2\pi fx}dxdy = e^{-j2\pi f\tau}\delta_{mn} . \tag{3.5}$$

Performing a change of variables with $\beta = f/y$ gives the equation

$$\int_0^\infty \int_{-\infty}^\infty \phi_m(\tau,x\, f/\beta) e^{-j2\pi f(x-\tau)} \sqrt{\beta/f}\, \frac{|f|}{\beta^2} S_n(\beta)\, dx\, d\beta = \delta_{mn} . \tag{3.6}$$

At this point, the orthogonality properties of the $s_n(t)$ play an important role in determining the algorithm. Let us assume that $S_n(f) = 0$, for $f < 0$, that

$$\int_0^\infty S_n(f) S_m^*(f) g(f)\, df = \delta_{mn} , \tag{3.7}$$

and that the set $\{s_n(t)\}$ is complete in this weighted L^2 space. Here, $g(f)$ is an appropriate nonnegative weighting function. Using (3.7) in (3.6), we get (after making the substitution $y = f/\beta$)

$$\phi_n(\tau,x,y) = \int_0^\infty \frac{|f|}{y^{3/2}} e^{j2\pi f(x-\tau)} g(f/y) S_n^*(f/y)\, df = \sqrt{y}\, \acute{s}_n^*(y(\tau-x)) , \tag{3.8}$$

where $\acute{s}_n(t)$ is the inverse Fourier transform of $|f| g(f) S_n(f)$. This gives a family of possible reconstruction algorithms determined by $g(f)$. Some choices of $g(f)$ give particularly nice properties for the reconstruction. The synthesis equation then takes the form

$$\sum_{n=1}^\infty \int_{-\infty}^\infty e_n(t) \sqrt{y}\, \acute{s}_n^*(y(t-x))\, dt . \tag{3.9}$$

In [1] the magnitude signs on $|f|$ are important because the transmitted signals are not represented in the analytic forms. Here, the magnitude signs are unnecessary because $S_n(f) = 0$ for $f < 0$. There is a question about whether (3.9) in fact equals $D(x,y)$ due to the fact that the signals $s_n(t)$ are designed to form an orthonormal set only in the space of functions whose Fourier transform is defined for positive frequencies. In particular, notice that the Fourier transform of $e_n(t)$ is

$$E_n(f) = \int_0^\infty \int_{-\infty}^\infty \frac{1}{\sqrt{y}} S_n(f/y) e^{-j2\pi fx} D(x,y)\, dx\, dy .$$

Thus, $E_n(f) = 0$ for $f < 0$. Since $D(x,y) = 0$ for $x < 0$, however, the Fourier transform of $D(x,y)$ in the x variable, $D_1(f,y)$,

$$D_1(f,y) = \int_0^\infty e^{-j2\pi fx} D(x,y)\, dx ,$$

has the property that its real part is the Hilbert transform in the f variable of its imaginary part. In the following, the convergence properties are studied for the set of reflectivity densities $D(x,y)$ such that $D_1(f,y) = 0$ for $f < 0$.

3.1. Naparst's First Algorithm

A method from the literature is now adapted to the situation where the analytic representation of the signals is used. Naparst [2] proposed a method for wideband imaging which can be derived from the above using $g(f) = 1$. For this choice, the synthesis functions are given by

$$\phi_n(t,x,y) = \sqrt{y}\, \acute{s}_n^*(y(t-x)) , \tag{3.10}$$

where $\acute{s}_n(t)$ is the inverse Fourier transform of $f S_n(f)$.

Note that multiplication by f in the frequency domain corresponds to taking a derivative in the time domain:

$$\hat{s}(t) = \frac{1}{2\pi j}\frac{d}{dt}s(t)\,. \tag{3.11}$$

This derivative operation enhances noise when $r_n(t)$ is substituted for $e_n(t)$ in (3.9). By a simple change of variables, each term in (3.9) may be shown to equal

$$\int \frac{1}{y^{3/2}}\hat{e}_n(t/y+x)s_n^*(t)dt\;, \tag{3.12}$$

where $\hat{e}_n(t)$ is the inverse Fourier Transform of $fE_n(f)$.

Equation (3.12) is somewhat disconcerting if there is additive noise, because the indicated operation includes taking a derivative. Since (3.9) is equivalent to (3.12) mathematically, this indicates a potential problem with the reconstruction. The actual derivative is taken on the transmitted signal, however, and may be done off-line, so the computation in (3.9) may not suffer the same problems that (3.12) apparently will.

3.2. New Wideband Algorithm

The simple arguments which gave rise to the concept of using a processor matched to the transmitted signal as in (2.6) motivates a choice of $g(f)$ such that the synthesis functions are in fact matched to the transmitted signal. That is, it is desired to have $\hat{s}_n(t) = s_n(t)$. This gives the choice of $g(f) = 1/f$. For this choice, the synthesis equation has optimal noise performance. This is explored further in the following sections.

4. Properties of the Wideband Algorithm

For any of the class of reconstruction algorithms proposed in the previous section, certain properties hold. These properties are important when considering the limitations of actual implementations of the algorithms. First, the wideband ambiguity function is defined and a few of its properties are outlined. These lead to a characterization of each term in the reconstruction (3.9) as a projection of $D(x,y)$. In addition, convergence of the reconstruction is obtained.

Define the wideband cross-ambiguity function to be

$$A_{sh}(x,y) = \int_{-\infty}^{\infty} \sqrt{y}\,s(y(t-x))h^*(t)dt\;. \tag{4.1}$$

In the following, we make use of the cross-ambiguity function between the transmitted signal and the signal used in the reconstruction. Define

$$\hat{A}_{nm}(x,y) = A_{s_n\hat{s}_m}(x,y)\,. \tag{4.2}$$

Substituting (3.4) into (3.9), an expression for each term in the reconstruction in terms of the wideband cross-ambiguity function is obtained as

$$\begin{aligned}\int e_n(t)\sqrt{y}\,\hat{s}_n^*(y(t-x))dt &= \iint\Big[\int \sqrt{y'}s_n(y'(t-x'))\sqrt{y}\,\hat{s}_n^*(y(t-x))dt\Big]D(x',y')dx'dy' \\ &= \iint \hat{A}_{nn}(y(x'-x),y'/y)D(x',y')dx'dy'\,.\end{aligned} \tag{4.3}$$

The first result in this section is that the terms (4.3) represent projections of $D(x,y)$. For the proof, see the appendix.

Property 1: The ambiguity functions (4.2) have the property

$$\iint \hat{A}_{nm}(y(x'-x),y'/y)\hat{A}_{kl}(y'(\xi-x'),\eta/y')dx'dy' = \delta_{nl}\hat{A}_{km}(y(\xi-x),\eta/y)\,. \tag{4.4}$$

This property gives the desired projection property by letting $n = m$ and $k = l$. Each term then gives the projection onto a linear subspace of the set of possible $D(x,y)$. If this space is equipped with an appropriate norm, the reconstruction may be shown to converge. The appropriate norm is determined by the weight function $g(f)$. The choice of $g(f) = 1/\sqrt{|f|}$ gives a particularly nice norm as discussed by Naparst [2]. For that choice, the cross-ambiguity functions between transmitted signals are orthonormal with an appropriate weight. Expanding $D(x,y)$ in terms of the cross-ambiguity functions then gives the convergence. For the other choices of $g(f)$ it is easier to address the convergence of the reconstruction directly. The proof of the following is in [1].

Property 2: Suppose that for some $D(x,y)$, (4.3) is zero for all n and all (x,y). Then $D(x,y) = 0$ almost everywhere.

From the reconstruction (3.9) and the projection property (4.4), the space in which the $\{D(x,y)\}$ exist is separated into disjoint subspaces. Each subspace is recovered exactly from one transmitted signal. A basis for each of these subspaces may be obtained using the transmitted signals. This also gives basis elements for the null space of the integral (3.4). The following analysis assumes that $g(f) = 1/f$. Assuming the functions $\{S_n(f)\}$ satisfy (3.7), the set $\{S_n(f)/\sqrt{f}\}$ is a complete orthonormal basis for $L^2(\mathbf{R})$. As discussed in Section 3, the space of reflectivity functions $D(x,y)$ considered are those for which $D_1(f,y) = 0$ for $f < 0$. Equipping this space with the inner product (where $\overline{D}(x,y)$ is any other reflectivity)

$$<D,\overline{D}> = \int_0^\infty\int_0^\infty D_1(f,y)\overline{D}_1^*(f,y)\,df\,dy$$

generates the L^2 space in which the functions B_{mn} in (4.5) are orthonormal.

Property 3: A complete set of orthonormal functions for the space of reflectivity functions just defined is given by the cross-ambiguity functions of $\{s_n(t)\}$ and the inverse Fourier Transforms of $\{S_n(f)/\sqrt{f}\}$:

$$B_{mn}(x,y) = \int_0^\infty S_m(f)S_n^*(f/y)e^{j2\pi fx}\frac{1}{\sqrt{yf}}df \ . \tag{4.5}$$

See the appendix for the proof. It is easily shown that $\{B_{lk} : k \neq n\}$ is a basis for the null space of (3.4). In particular, the change of variables $y' = f/y$ gives

$$\int_0^\infty\int_{-\infty}^\infty \sqrt{y}\, s_n(y(t-x))B_{lk}(x,y)\,dx\,dy = \delta_{nk}\int_0^\infty S_l(f)\frac{1}{\sqrt{f}}e^{j2\pi ft}\,df \ . \tag{4.6}$$

Notice that the right hand side of (4.6) is one of the orthonormal basis functions. Thus, (3.4) maps orthonormal basis functions in the space of reflectivity functions to orthonormal basis functions in $L^2(\mathbf{R})$. This is important for the results in Section 5, and this choice of $g(f)$ is the only one for which this holds.

5. Estimation Theoretic Applications

There have been several papers in the literature discussing applications of estimation theory to narrowband radar imaging [13,15,25]. In [15] more general estimation problems are introduced. Also in [15] a fair amount of emphasis is placed on problems which, when properly discretized, yield unitary mappings from the image space to the measurement space. In this section, that assumption is justified by showing that in the continuous problem, the mapping is unitary. One important consequence of the derivation is the one-to-one correspondence between a countable set of white noise functions and a single two dimensional function which is white. A discussion of this problem for the narrowband case is contained in [1].

Suppose that the received signals $\{e_n(t)\}$ are observed in complex white Gaussian noise, as in (2.1). The signals $r_n(t)$ are used in the synthesis equations instead of $e_n(t)$. The reconstruction has two components. One is due to the signal, the second is due to the noise and is denoted $v(x,y)$

$$v(x,y) = \sum_{n=1}^{\infty} \frac{1}{\sqrt{2}} \int_{-\infty}^{\infty} \sqrt{y}\, s_n^*(y(t-x)) w_n(t)\, dt \ . \tag{5.1}$$

$v(x,y)$ is white when the elements of the set $\{S_n(f)\}$ satisfy (3.7) with $g(f) = 1/f$.

Property 4: The spatial process $v(x,y)$ is white with spectral height N_0.

This property is very important for detection and estimation applications. It indicates that, for large data sets, given any model for $D(x,y)$, the synthesis equation yields $D(x,y)$ plus a white Gaussian random field. The white noise signals $w_n(t)$ may be viewed as coming from the white random field $v(x,y)$ using the analysis equation,

$$w_n(t) = \int_{0}^{\infty}\int_{-\infty}^{\infty} \sqrt{y}\, s_n(y(t-x)) v(x,y)\, dx dy \ . \tag{5.2}$$

In order to motivate the use of Property 4 for estimation problems, a specific problem is addressed. Suppose that $D(x,y)$ is a spatially white Gaussian random process with scattering function $\Theta(x,y)$. Following [26, p. 395] the integral of $\Theta(x,y)$ over any region may be recovered from $m(x,y) = D(x,y) + v(x,y)$. In particular, look at the rectangle with corners at (x,y) and $(0,0)$, for $x > 0$ and $y > 0$. Define a partition of this rectangle to have elements $\Pi_{N,k}$, where N is the number of elements in the partition and k is the partition index. Then, the set $\{m_{N,k}\}$

$$m_{N,k} = \iint_{\Pi_{N,k}} m(x,y)\, dx dy \tag{5.3}$$

are independent complex valued Gaussian random variables with variances

$$E[\,|m_{N,k}|^2] = \iint_{\Pi_{N,k}} (\Theta(x,y) + N_0)\, dx dy \ . \tag{5.4}$$

Assuming that the $N+1$ partition is formed from the N partition by bisecting a partition element, the proof from Doob [26, pp. 395-396] applies and

$$\lim \sum_{k=1}^{N} |m_{N,k}|^2 = \int_{0}^{y}\int_{0}^{x} (\Theta(x',y') + N_0)\, dx' dy' \tag{5.5}$$

with probability one, where the limit is taken as the maximum partition element's area goes to zero.

Equation (5.5) gives an algorithm for obtaining the integral of $\Theta(x,y)$ over a patch. The size of the patch and the location in the (x,y) plane of the patch are irrelevant. Thus, the integral of the scattering function over arbitrarily small patches can be recovered with probability one.

Other numerically attractive approaches for recovering $\Theta(x,y)$ are given in [15].

6. Conclusions

The algorithms presented in Section 3 for wideband radar imaging are based upon viewing the received radar data as being a transformation of the desired image data. The special property which the new wideband algorithm ($g(f) = 1/f$) presented in Section 3.3 has is that the set of cross-ambiguity functions (4.5) form an orthonormal basis for the space of reflectivity functions. Furthermore, the analysis equation (3.4) and the synthesis equation (3.9) map orthonormal basis functions for an infinite number of copies of $L^2(\mathbf{R})$ to orthonormal basis functions for the space of reflectivity functions. That is, as shown in (4.6) and the appendix,

$$\int_0^{\infty} E_n(f) S_k^*(f) \frac{1}{\sqrt{f}} df = \int_0^{\infty}\int_{-\infty}^{\infty} D(x,y) B_{kn}^*(x,y) dx dy \ . \tag{6.1}$$

This result is central to the proof that the reconstruction algorithms map a countable set of white Gaussian noise functions to a single two-dimensional white Gaussian noise function.

This paper has addressed some of the important practical considerations which must be taken into account, but has not addressed others. For example, only a finite number of signals are used in an actual application. The projection property introduced in Property 1 describes the limitations on the possible reconstruction. The set $\{B_{km}(x,y) \mid m \neq n\}$ forms a basis for the set of ambiguity functions in the null space of the n^{th} projection (4.3). The reconstructions proposed ignore the effect of target motion during the illumination interval. If constraints on the reconstruction such as finite target extent are introduced, improvements in the reconstruction may be obtained.

Signal design may be partially addressed using the methods in this paper. For example, if a certain target is expected, the signal may be designed so that this target is substantially in the range space of one of the projections.

Appendix

Proof of Property 1: The wideband cross-ambiguity function may be found in the frequency domain by the formula

$$\hat{A}_{nm}(x,y) = \int_0^{\infty} \frac{1}{\sqrt{y}} S_n(f/y) e^{-j2\pi fx} f g(f) S_m^*(f) df = \int_0^{\infty} S_n(f) \mid yf \mid g(yf) \sqrt{y} S_m^*(yf) e^{-j2\pi fyx} df \ .$$

Using these expressions in (4.4), we obtain

$$\int_0^{\infty}\int_{-\infty}^{\infty} \hat{A}_{nm}(y(x'-x), y'/y) \hat{A}_{kl}(y'(\xi-x'), \eta/y') dx' dy'$$

$$\int_0^{\infty}\int_{-\infty}^{\infty}\int_0^{\infty}\int_0^{\infty} S_n(f) S_m^*(y'f/y) g(y'f/y) S_k(y'f'/\eta) g(f') S_l^*(f') \left[\frac{y'}{y}\right]^{3/2} \left[\frac{y'}{\eta}\right]^{1/2} f f' e^{-j2\pi fy'(x'-x)} e^{-j2\pi f'y'(\xi-x')} df\, df'\, dx$$

The integral over x' gives an impulse $\delta(y'(f-f'))$. Then the integral over f' collapses to a factor of $1/y'$ and setting $f = f'$ giving

$$\int_0^{\infty}\int_0^{\infty} S_n(f) S_l^*(f) g(f) S_k(y'f/\eta) S_m^*(y'f/y) g(y'f/y) \left[\frac{\eta}{y}\right]^{3/2} e^{-j2\pi fy'(\xi-x)} \frac{f^2 y'}{\eta^2} df\, dy' \ .$$

Finally, performing the change of variables $f' = y'f/\eta$, $dy' = \eta df'/f$ gives

$$\int_0^{\infty} S_n(f) S_l^*(f) g(f) df \int_0^{\infty} S_k(f') S_m^*(\eta f'/y) g(\eta f'/y) \mid f' \mid \left[\frac{\eta}{y}\right]^{3/2} e^{-j2\pi f'\eta(\xi-x)} df'$$

This is the desired result. □

Proof of Property 3: To see that this is an orthonormal set, evaluate

$$<B_{mn}, B_{lk}> = \int_{0}^{\infty}\int_{-\infty}^{\infty} B_{mn}(x,y) B_{lk}^{*}(x,y)\,dx\,dy = \int_{0}^{\infty}\int_{0}^{\infty} S_k(f/y) S_n^{*}(f/y) S_m(f) S_l^{*}(f) \frac{1}{yf}\,df\,dy \ .$$

Performing a change of variables, $y' = f/y$ gives

$$<B_{mn}, B_{lk}> = \left[\int_{0}^{\infty} S_k(y') S_n^{*}(y') \frac{1}{y'}\,dy'\right]\left[\int_{0}^{\infty} S_m(f) S_l^{*}(f) \frac{1}{f}\,df\right]$$

$$= \delta_{ml}\,\delta_{nk} \ .$$

To see that the set $\{B_{mn}\}$ is complete, first notice that the set $\{S_m(f) S_n^{*}(y')/\sqrt{f}\}$ spans $L^2(\mathbf{R}_{+}\times\mathbf{R}_{+}, df\,dy'/y')$. Then, changing variables, the set $\{S_m(f) S_n^{*}(f/y)/\sqrt{fy}\}$ spans $L^2(\mathbf{R}_{+}\times\mathbf{R}_{+}, df\,dy)$. □

Proof of Property 4: Notice that the random variables

$$w_{nm} = \int_{-\infty}^{\infty} s_m^{*}(t) w_n(t)\,dt$$

are i.i.d. complex Gaussian with variance N_0. By Property 3, $s_m(t)$ is mapped through the synthesis equation to $B_{mn}(x,y)$, and

$$w_{nm} = \int_{0}^{\infty}\int_{-\infty}^{\infty} B_{mn}^{*}(x,y) v(x,y)\,dx\,dy \ . \ \square$$

References

1. J. A. O'Sullivan, "A Transform Approach to Radar Imaging," *submitted to IEEE Transactions on Information Theory*, July 1991.
2. H. Naparst, "Dense Target Signal Processing," *IEEE Transactions on Information Theory*, vol. IT-37, pp. 317-327, March 1991.
3. H. Naparst, "Radar Signal Choice and Processing for a Dense Target Environment," *in Signal Processing, Part II: Control Theory and Applications, F. A. Grunbaum, J. W. Helton, and P. Khargonekar, eds.*, vol. IMA - 23, pp. 293-319, Springer-Verlag, New York, 1990.
4. W. Miller, Jr., "Topics in Harmonic Analysis with Applications to Radar and Sonar," *in Radar and Sonar, Part I, R. E. Blahut, W. Miller, Jr., and C. H. Wilcox, eds.*, vol. IMA - 32, Springer-Verlag, New York, 1991.
5. P. Maass, "Wideband Radar: the Hyp Transform," *Inverse Problems*, vol. 5, pp. 849-857, 1989.
6. L. Auslander and I. Gertner, "Wide-Band Ambiguity Function and *ax + b* Group," *in Signal Processing, Part I: Signal Processing Theory, L. Auslander, T. Kailath, and S. Mitter, eds.*, vol. IMA - 22, pp. 1-12, Springer-Verlag, New York, 1990.
7. C. H. Wilcox, "The Synthesis Problem for Radar Ambiguity Functions," *in Radar and Sonar, Part I, R. E. Blahut, W. Miller, Jr., and C. H. Wilcox, eds.*, vol. IMA - 32, Springer-Verlag, New York, 1991.
8. W. Schempp, "Radar Ambiguity Functions, the Heisenberg Group, and HolomorphicTheta Series," *Proceedings American Mathematical Society*, vol. 92, pp. 103-110,

1984.

9. L. Auslander and R. Tolmieri, ''Radar Ambiguity Functions and Group Theory,'' *SIAM Journal on Mathematical Analysis*, vol. 16, pp. 577-599, 1985.
10. I. Daubechies, ''The Wavelet Transform, Time-Frequency Localization and Signal Analysis,'' *IEEE Transactions on Information Theory*, vol. IT-36, pp. 961-1005, Sept. 1990.
11. C. Heil, ''Wavelets and Frames,'' *in Signal Processing, Part I: Signal Processing Theory, L. Auslander, T. Kailath, and S. Mitter, eds.*, vol. IMA - 22, pp. 147-160, Springer-Verlag, New York, 1990.
12. A. Grossman and J. Morlet, ''Decomposition of Hardy Functions into Square Integrable Wavelets of Constant Shape,'' *SIAM Journal on Mathematical Analysis*, vol. 15, pp. 723-736, July 1984.
13. D.L. Snyder, J.A. O'Sullivan, and M.I. Miller, ''The Use of Maximum-Likelihood Estimation for Forming Images of Diffuse Radar-Targets from Delay-Doppler Data,'' *IEEE Transactions on Information Theory*, vol. IT-35, pp. 536-548, 1989.
14. P. Moulin, J. A. O'Sullivan, and D. L. Snyder, ''A Sieve-Constrained Maximum-Likelihood Method for Target Imaging,'' *to appear in Radar and Sonar, Part II*, Springer-Verlag, New York, 1991.
15. P. Moulin, J. A. O'Sullivan, and D. L. Snyder, ''A Method of Sieves for Multiresolution Spectrum Estimation and Radar Imaging,'' *submitted to IEEE Transactions on Information Theory*, 1991.
16. H. L. Van Trees, *Detection, Estimation, and Modulation Theory, Part III,* John Wiley and Sons, New York, 1971.
17. J. Shapiro, B.A. Capron, and R.C. Harney, ''Imaging and Target Detection with a Heterodyne-Reception Optical Radar,'' *Applied Optics*, vol. 20, No.19, pp. 3292-3313, 1981.
18. M. Bernfeld, ''CHIRP Doppler Radar,'' *Proc. IEEE*, vol. 72, No. 4, pp. 540-541, April 1984.
19. E. Feig and F. A. Grunbaum, ''Tomographic Methods in Range-Doppler Radar,'' *Inverse Problems*, vol. 2, pp. 185-195, 1986.
20. E. Feig and F. P. Greenleaf, ''Inversion of an Integral Transform Associated with Tomography in Radar Detection,'' *Inverse Problems*, vol. 2, pp. 405-411, 1986.
21. D. L. Mensa, *High Resolution Radar Imaging,* Artech House, Norwood, MA, 1981.
22. D. C. Munson, Jr. and R. L. Visentin, ''A Signal Processing View of Strip-Mapping Synthetic Aperture Radar,'' *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-37, pp. 2131-2147, Dec. 1989.
23. J. P. Fitch, *Synthetic Aperture Radar,* Springer-Verlag, New York, 1988.
24. D. R. Wehner, *High Resolution Radar,* Artech House, Dedham, MA., 1987.
25. J. A. O'Sullivan and D. L. Snyder, ''High Resolution Radar Imaging Using Spectrum Estimation Methods,'' in *Signal Processing Part II: Control Theory and Applications*, ed. F. A. Grunbaum, J. W. Helton, and P. Khargonecker, pp. 335-346, Springer-Verlag, New York, 1990.
26. J. L. Doob, *Stochastic Processes,* John Wiley and Sons, New York, 1953.

PERFORMANCE OF MAXIMUM LIKELIHOOD METHOD FOR DIRECTION OF ARRIVAL ESTIMATION OF WIDEBAND SIGNALS[+]

R. L. Kashyap and S. M. Kattepur
School of Electrical Engineering, Purdue University
West Lafayette, IN 47907-1285

Abstract

Direction of arrival (DOA) estimation of wideband signals and the spectral density estimation by Maximum Likelihood (ML) method and its performance in comparison to other existing methods is considered in this paper. The unique feature of this ML method is that it does not need the knowledge of the true spectral density matrix as far as the DOA estimation is considered. However, the spectral density matrix which is needed for the DOA estimation is parameterized by a finite order AR model, and a new method to estimate these AR parameters is developed. The assumption on the order of the data is also not needed for the estimation of the angles of arrival. This ML estimate can also handle the sources with identical spectra. The simulation results under various scenario are shown to be in accordance with the theory developed by Kashyap [1]. The ML method is shown to be consistent and perform very well with less number of snapshots, unlike the currently existing methods which require a large number of snapshots and thereby involve more computation. Performance of ML is also shown to be superior with respect to resolution capability, performance for correlated signals even under low Signal to Noise Ratios (SNR).

I. Introduction

There are several estimation methods for the direction of arrival of wideband signals. All these estimates perform very well when the M (# sensors) or N (# snapshots) is large, the signal to noise ratio is large, and the DOAs are relatively far apart. It is only under stringent conditions like low M or N or SNR or correlated signals, the differences between the accuracy of ML estimates and others become pronounced. The ML estimation for multiple sources was formulated in [2,14]. Previously it was thought that only under idealized conditions, i.e., when the number of sources and the source spectral density matrix are known, ML estimation is a manageable multivariate nonlinear optimization problem [10] and [2,4,9,11].

Recently Kashyap [1] has developed a different version of Maximum Likelihood for the DOA estimation of narrowband and wideband signals. We show that most of the theoretical results developed in [1] are meaningful in the practical sense. Our ML needs less number of snapshots than most of the other existing methods. The special feature of this ML is that it does not need the true value of the spectral density matrix for the DOA estimation. The spectral density is parameterized by a finite order AR model. Then it becomes a known function of these AR parameters. The procedure for estimating these AR parameters is also considered in this paper. We have taken second order AR model in our simulations for convenience. However, the knowledge of the correct model of the data is not a crucial factor for the purpose of estimating the angles of arrival. We show that ML can also handle the multiple sources with identical spectra.

The organization of the paper is as follows. The data model and the problem formulation for Maximum Likelihood procedure is considered briefly in section II, followed by the method of estimating the spectral density in the next section. Section IV gives some of the simulation results, and comparison of the ML method with currently popular methods like signal subspace [4], eigenstructure [6], coherent signal subspace [5,7], etc.

II. Problem Formulation

We give a brief explanation of the data model here. Details are in [1]. There are D broadband sources $s_k(.)$, $k = 1,...,D$, and M sensors. $u_m(.)$ are the IID Gauss $(0,\rho)$ random

---

[+] This work is supported in part by the Innovative Science and Technology (IST) Program of SDIO monitored by Office of Naval Research under Contract ONR N00014-85K-0611.

variables.

Let the observation received by m^{th} sensor be $y_m(t)$.

$$y_m(t) = \sum_{k=1}^{D} s_k(t-(m-1))\phi_k) + u_m(t), \quad t = 1,...,N \text{ and } m = 1,...,M \tag{1}$$

ϕ_k are the parameters containing the direction of arrival (DOA) information. These are the phase delays between the sensors expressed. We estimate ϕ_ks in most of our simulations but when the comparison needs to be done with other methods we consider the actual angle of arrival in degrees. There are N observations or snapshots. The DFT of $\{y_m(t), t = 1,2,....,N\}$ is given by $\{Y_m(n), n = 0,...,N\text{-}1\}$ where

$$Y_m(n) = \sum_{t=0}^{N-1} e^{-i\,2\pi nt/N}\, y_m(t), \quad n = 0,...,N-1$$

$$= \sum_{k=1}^{D} e^{-i(m-1)\omega_n\phi_k} S_k(n) + U_m(n)$$

where $S_k(n)$ and $U_m(n)$ are DFT of $s_k(t)$ and $u_m(t)$, respectively. $\omega_n = 2\pi n/N$. Let

$$Y(n) = \text{col}[Y_1(n), \ldots, Y_M(n)], \;\; S(n) = \text{col}[S_1(n), \ldots, S_D(n)], \;\; U(n) = \text{col}[U_1(n),...,U_M(n)]$$

$$Y(n) = A_n S(n) + U(n)$$

Let $A_n = [f_n(\phi_1), f_n(\phi_2), \ldots, f_n(\phi_D)]$,

$f_n(\phi) = \text{Col.}[1, e^{-i\omega_n\phi}, e^{-i\omega_n 2\phi}, \ldots, e^{-i\omega_n(M-1)\phi}]$

$$C_n = E[Y(n)Y^*(n)] = [A_n\Gamma(\omega_n)A_n^* + \rho I]N$$

$$\Gamma(\omega_n) = [\text{spectral density matrix of } s(.) \text{ at frequency } \omega_n = 2\pi n/N] = \Gamma_n.$$

It is well known that $Y(n)$, $n = 1,...,N$ are IID with Gauss density $(0,C_n)$.

$$p(\{Y.\} \mid \phi,\Gamma,\rho) = \prod_{n=0}^{N-1} 2\pi^{-M/2} \,|\det C_n|^{-1/2} \exp\left[-\frac{1}{2}Y^*(n)C_n^{-1}Y(n)\right]$$

$$\ln p(\{Y.\} \mid \phi,\Gamma,\rho) = -\frac{NM\ln 2\pi}{2} - \frac{1}{2}\sum_{n=0}^{N-1} [\ln\det C_n + \text{trace}(C_n^{-1}Y(n)Y^*(n))] \tag{2}$$

We have to maximize the above likelihood function with respect to the unknowns. The unknowns include not only the DOA's $\phi_1, \phi_2,...$ which are of interest to us, but also the nuisance variables $\Gamma_1, \Gamma_2,...$ etc., which are functions of the spectral density of the sources.

It is absolutely imperative that the total number of unknowns be made as small as possible. Hence we will model each of the sources by an autoregressive model of appropriate order (two in our simulation). In this case all the covariance matrices $\Gamma_1, \Gamma_2, \ldots, \Gamma_{N-1}$ are functions of these parameters of the AR model, whose number is much less than N.

We first estimate the parameters of the various AR models of the sources using the observations of one sensor. In this case the DOA parameters do not come into the picture - we have developed a new method, a maximum likelihood method, to do this. It is described in the appendix.

Using these estimated autoregressive parameters, we compute their correlation matrices $\{\Gamma_i\}$, substitute them in the likelihood expression (2) and minimize it wrt the DOA's ϕ_1, $\phi_2, \ldots, \phi_D$ and ρ. The minimization problem is straightforward.

Estimation of the spectral density AR parameter from $\{y(.)\}$ is computationally time consuming. We need to ask whether very accurate estimates of the Γ parameters are needed at all. Recall that they are "nuisance" parameters. This question has been addressed in great detail in this paper. We explore several computationally simple alternatives described later and we have found that the estimates of the DOA's using these simple methods of estimation give results that are as good as the DOA estimates obtained by using the more complicated estimates of Γ_i, as long as SNR is not too low (greater than −5db).

Summing up, our computation is in two steps. The first step is to obtain estimates of the Γ_i via the spectral density parameters of the signals. The second step is using these estimates in (2) and minimizing it to find DOA's.

III. Simulation Results and Comparison with Other Methods

We have conducted different experiments to study the properties of our ML method and compared with the performance of the other methods. We quote here only the results of a few of our experiments.

The performance of all the methods are compared using the estimated root mean square error (RMSE) of the direction of arrival ϕ_1.

$$\text{Estimated RMSE} = \frac{1}{t} \sum_{k=1}^{t} (\phi_1^k - \phi_1^0)^2$$

where t is the number of trials, ϕ_1^0 is the true value of DOA, $\phi_1^{(k)}$ is the estimate of ϕ_1^0 in the kth trial. Most of our experiments deal with two sources with DOA's ϕ_1^0 and ϕ_2^0. We give the plots of RMSE of only one of the DOA's because the other one is similar.

$$\text{Estimated Bias of estimate} = \frac{1}{t} \sum_{k=1}^{t} |\phi_1^{(k)} - \phi_1^0|$$

A. Unknown spectral density of signals

In our experiment, there are two sources, each of which obeys an AR(4) model. We compute the DOA estimates under 3 conditions:

Case 1) We compute the correct Γ for the known AR(4) parameters, substitute them in (1) and minimize the likelihood.

Case 2) Represent the source as an AR(2) model, estimate the 3 parameters of AR(2) model using the observations of one sensor and the algorithm in the appendix.

Case 3) Represent the source as an AR(2) model, but choose any *arbitrary values* for the coefficients of the AR model as long as they are within the stationary region (the triangle region [19]), i.e., the region in which the model will be stationary.

We plot the RMSE of one DOA ϕ_1 versus the SNR for all the cases in figure 1. For each SNR value, each point is the average of the 20 values obtained from 20 independent runs. Note that the graphs of 1,2, and 3 are close to each other when the SNR is not too small (i.e., greater than −5db). The correct knowledge of the AR parameters becomes important only at very small SNR like −10db.

The surprising conclusion is that a rough estimate of the Γ's parameters obtained via an AR(2) model is sufficient (even when source is AR(4)) (Case 3) when SNR is not too low. Note our experiment does *not* say that we can use any arbitrary "Γ_i" values.

To further validate our conclusion, we consider the case of one source which obeys an AR(2). We obtained the likelihood expression for the observations of one sensor only (eq. 7 in appendix) and plotted them as a function of the 2 AR parameters. The 3D plot is fairly smooth without any sharp peaks or valleys.

B. Bias and RMSE as function of N

The ML estimate is biased. In figure 2 we have plotted the log |BIAS| and log RMSE with respect to log N, N being # snapshots. Note BIAS is the dominant contributor for RMSE, the contribution of standard deviation is small. Both RMSE and BIAS are of order $(O(1/\sqrt{N})$ as explained in [1].

C. Comparison with signal subspace method of Su and Morf [4]

We compare our ML results with the results of Signal Subspace method (SS) by Su and Morf method. In both the methods, 20 trials are considered for different signal to noise ratios from 20 dB to −10 dB. In both the methods, sources have second order AR spectra with the following spectral factors:

$$\text{True spectra for Source 1: } \frac{1}{1+0.371z^{-1}+0.36z^{-2}}$$

$$\text{Source 2: } \frac{1}{1+0.433z^{-1}+0.49z^{-2}}$$

Both the sources are chosen as highly overlapping spectra with the same center frequencies of 0.3 Hz. The number of sensors is the same in both cases, namely 4. The two sources in SS method are 4 degrees apart (88 and 92). For all practical purposes this situation could be thought of as estimation of two sources separately. We consider a harder case with the true DOAs as 86.9 and 87.26 degrees, which are 2.36 degrees apart.

The SS method needs 500 snapshots, whereas in our method 128 snapshots are enough. The RMSE of one of the sources (ϕ_1) are plotted in figure 3 for both the ML and the SS method. The plots are taken for only one of the two sources as the plot for the other source is also similar. The superiority of the ML method is striking, especially at low SNR. At −5db, RMSE with ML method is about one third of the value with SS method. Results are in figure 3.

D. Comparison of ML with CSS [5,7]

Coherent signal subspace (CSS) of Wang and Kaveh [5,7] can estimate DOA of coherent signals. We compare ML and CSS under similar conditions. CSS method [5,7] uses 16 sensors (M = 16) and 2112 (64x33) snapshots (N = 2112). In CSS the sources are 3 degrees apart, where in ML they are 1.04 degrees apart. We give ML results with both N = 2112 and N = 128. In figure 4, the RMSE for all the 3 cases (ML with N = 128, ML with N = 2112, CSS with N = 2112). Above −5db, ML with N = 128 outperforms CSS with N = 2112. Results are in figure 4.

E. Correlated sources.

In the multipath environment, coherency of the signals is a common feature. When r, the correlation coefficient between the signals, is one, the signals are fully correlated or coherent. Some of the methods like Eigen structure methods [6], incoherent Signal Subspace methods [9] can not handle highly correlated sources. The results are in figure 5. The performance of our ML is not greatly effected by the correlation. We have considered a correlation coefficient of 0.7 to illustrate this fact. There is not much variation in the results.

F. Comparison with other methods

In table 1, we list the number of snapshots (N) and the number of sensors (M) in the various methods so as to achieve resolution of two closely spaced sources. We have listed most of the well known methods like the signal subspace [4], the coherent signal subspace [5,7] and its extensions [8] and several other methods [13,16,18].

Porat and Friedlander have proposed a method for the joint estimation of spatial and spectral parameters for multiple sources with broad or narrowband spectra [13]. A parameter model is derived for the multichannel measurement vector. The model parameters are the time delays and the autoregressive moving average parameters of all the sources. An ARMA (Auto Regressive Moving Average) model with a spectral structure is used to represent the sensor outputs. The parameters of that model are the relative source/sensor time delays and the AR or ARMA spectral parameters of each source. The assumptions made here are that the sources do not have identical spectra and that the source spectra are pure autoregressive and of first or second order. However, this assumption is relaxed and modifications have been suggested to handle such cases. We do not need any of the above assumptions or modifications for ML to estimate DOA of the signals. In our ML, identical spectrum does not affect the DOA estimation as the relative phase delays will be different for different DOAs. We fit second order AR model for the data in order to evaluate the spectral density matrix and the values of the AR parameters could either be estimated or approximately taken so that they satisfy the condition for stability. The number of snapshots used by Porat and

Friedlander is 1000 which is very high compared to the one used by us. The former method cannot resolve the two sources whose SNR are below –9db and the noticeable feature in their simulations is that they have always taken the two sources with a 6 db difference in their SNRs. Perhaps the method fails to resolve the two sources of equal energy. This method first estimates the spectral parameters and then the time delays. On the other hand, ML does not involve much computation as the spectral parameters need not be estimated for the estimation of DOA and the number of snapshots and sensors required are very low compared to all the other methods.

Simple generalizations of narrowband algorithms to the wideband problems have some problem because the signal subspace for one band is different from that of another band, preventing a stronger separation of well defined signals and noise spaces. In procedures described in [4], the final estimate will not resolve DOAs that are already not resolved by the spectral estimates of the individual bands. In one of the simulation results they have considered 3 sources located at 36, 60 and 120 degrees which can not be considered as closely spaced sources at all. As we have already shown, ML can resolve two very closely spaced sources. We have also tried to resolve the two sources (29.56 and 28.52 degrees) with $M = 4$ and $N = 128$ when they are fully correlated. We can resolve them under high SNR case. However, below 5 dB it is not possible to resolve them. By increasing the number of sensors it could be resolved.

The spectral method proposed by Nawab, et al. [18] has very low resolving power compared to ML and CSS method [5]. The method uses the zero delay covariances of the complex analytic representations of the sensor signals of a planar array to estimate the wave number spectrum. From the spectrum, source bearings are determined over the entire 360 range. The computational requirements of this method are of the same order as frequency wave number analysis on a planar array at a single frequency. Tri delta microphone array configuration is used with 9 sensors and the number of snapshots used is 4096. At low signal to noise ratios resolving two sources would be difficult for this spectral estimation method as false peaks appear in the spectrum. Although the computation is less in this method, the accuracy of the estimates is poor.

Buckley and Griffiths [8] have improved the CSS method [5,7] and have introduced a method (BASS - ALE method) which derive estimates directly from broadband data covariance matrices unlike the narrowband decomposed Signal Subspace methods. In simulations, the true covariance matrix is assumed to be known. A flat spectral density is assumed. Our ML does not need the true knowledge of the spectral density matrix for the DOA estimation. We have conducted some experiments to show that the performance of ML is unperturbed when the spectral density matrix deviates from its true value (Figure 1). Like all the other Eigen based methods BASS - ALE method also needs very large number of snapshots. However, its resolution is more than CSS method but involves more computation compared to CSS method. Results indicate that BASS - ALE estimator gives very small or zero spectral peak position BIAS unlike CSS method.

The spatiotemporal eigenstructure method [6] fails to resolve the two source under low SNR conditions.

Method	# Snapshots (N)	# Sensors (M)
ML (This paper)	128 (or less N)	4 (or less 3)
SS by Su and Morf [4]	500	4
Porat & Friedlander [13]	1000	2
CSS by Wang et al. [5-7]	2112	16
BASS-ALE [8]	2112	16
Nawab et al. [18]	4096	9
Eigen structure [6]	6400	5

Table 1. # snapshots (N) and # sensors (M) needed in various methods to resolve 2 close sources.

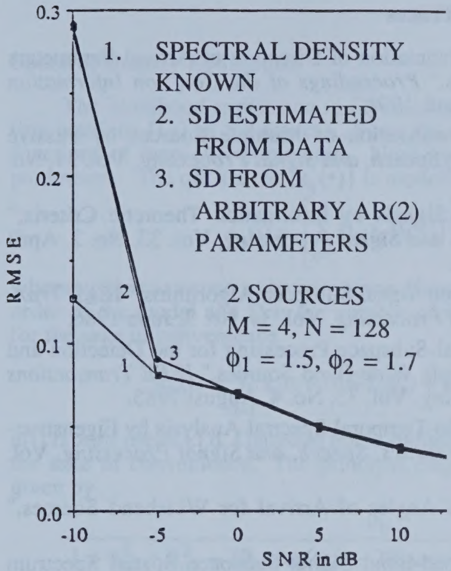


Figure 1

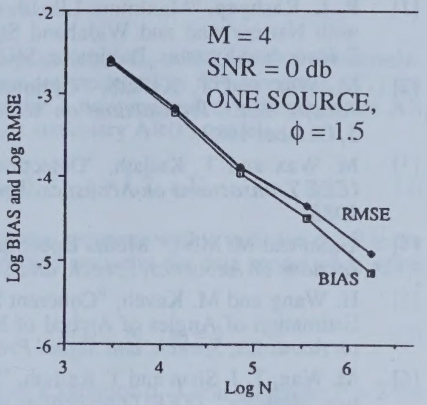


Figure 2

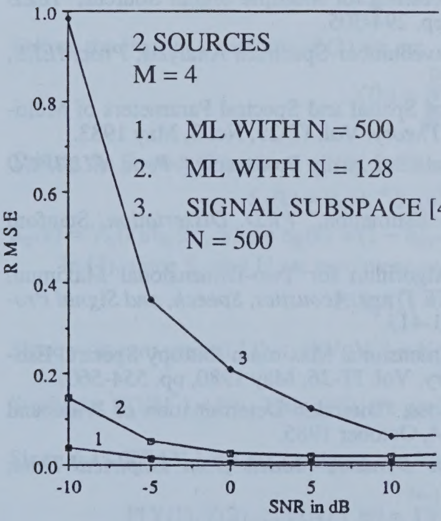


Figure 3. Comparison

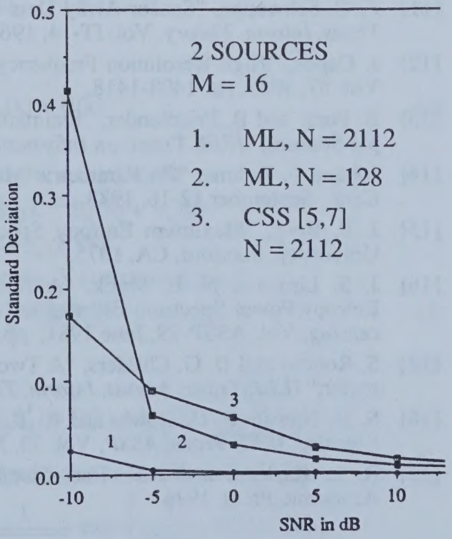


Figure 4. Comparison

VI. References

[1] R. L. Kashyap, "Maximum Likelihood Estimation of Direction of Arrival Parameters with Narrowband and Wideband Signals," *Proceedings of the Conf. on Information Science and Systems,* Baltimore, MD, March 1991.

[2] M. Wax and T. Kailath, "Optimum Localization of Multiple Sources by Passive Arrays," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. 31, No. 5, October 1989.

[3] M. Wax and T. Kailath, "Detection of Signals by Information Theoretic Criteria," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. 33, No. 2, April 1985.

[4] G. Su and M. Morf, "Modal Decomposition Signal Subspace Algorithms," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. 34, No. 3, June 1986.

[5] H. Wang and M. Kaveh, "Coherent Signal-Subspace Processing for the Detection and Estimation of Angles of Arrival of Multiple Wide-Band Sources," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. 33, No. 4, August 1985.

[6] M. Wax, T. J. Shan and T. Kailath, "Spatio-Temporal Spectral Analysis by Eigenstructure Methods," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. 32, No. 4, Aug 1984.

[7] H. Wang and M. Kaveh, "Estimation of Angles of Arrival for Wideband Sources," *Conf., IEEE,* 1984.

[8] K. M. Buckley and L. J. Griffiths, "Broad-Band Signal Subspace Spatial Spectrum (BASS-ALE) Estimation," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. 36, No. 7, July 1988.

[9] G. Su and M. Morf, "The Signal Subspace Approach for Multiple Wide Band Emitter Location," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. 31, No. 6, December 1983.

[10] J. F. Bohme, "Estimation of Source Parameters by Maximum Likelihood and Nonlinear Regression," *Conf., IEEE,* 1984.

[11] F. C. Schweppe, "Sensor Array Data Processing for Multiple Signal Sources," *IEEE Trans. Inform. Theory,* Vol. IT-14, 1968, pp. 294-305.

[12] J. Capon, "High Resolution Frequency-wavenumber Spectrum Analysis, *Proc. IEEE,* Vol. 57, 1969, pp. 1408-1418.

[13] B. Porat and B. Friedlander, "Estimation of Spatial and Spectral Parameters of Multiple Sources," *IEEE Trans. on Information Theory,* Vol. IT-29, No. 3, May 1983.

[14] Johann F. Bohme, "On Parametric Methods For Array Processing," *Proc. EUSIPCO Conf.,* September 12-16, 1983.

[15] J. P. Burg, "Maximum Entropy Spectral Estimation," *Ph.D. Dissertation,* Stanford University, Stanford, CA, 1975.

[16] J. S. Lim and N. L. Malik, "A New Algorithm for Two-Dimensional Maximum Entropy Power Spectrum Estimation," *IEEE Trans. Acoustics, Speech, and Signal Processing,* Vol. ASSP-29, June 1981, pp. 401-413.

[17] S. Roucos and D. G. Childers, "A Two-Dimensional Maximum Entropy Spectral Estimator," *IEEE Trans. Acoust. Inform, Theory,* Vol. IT-26, May 1980, pp. 554-560.

[18] S. H. Nawab, F. U. Dowla and R. T. Lacoss, "Direction Determination of Wideband Signals," *IEEE Trans. ASSP,* Vol. 33, No. 4, October 1985.

[19] R. L. Kashyap and A.R. Rao, *Stochastic Dynamic Models from Empirical Data,* Academic Press, 1976.

APPENDIX

Spectral Density Estimation

The likelihood expression in (2) involves the parameters C_n which involve the correlation matrices $\{\Gamma_i\}$ of the sources. Since the sources are modelled by AR processes, $\{\Gamma_i\}$ are functions of these AR parameters. Now we consider a new method to estimate these AR parameters. The qth source $\{s_q(\bullet)\}$ is modelled by a stationary AR(P) model:

$$s_q(t) = \left[\sum_{i=1}^{P} \theta_{qi} s_q(t-i)\right] + v_q(t), \qquad q = 1,2,\ldots,D \tag{1}$$

where $v_q(t)$ is assumed to be zero mean Gaussian noise sequence with a variance ρ_q, P is the order of the model and D is the number of sources and we write the data model once again for the sake of convenience

$$y_m(t) = \sum_{q=1}^{D} s_q(t-(m-1)\phi_q) + u_m(t), \qquad t = 1,\ldots,N, \quad m = 1,\ldots,M \tag{2}$$

u(t) is zero mean i.i.d. Gaussian noise sequence with a variance ρ. Let us consider $P = 2$ for the sake of convenience. The principal diagonal elements of the spectral density matrix is given by

$$\frac{\rho_q}{1 + \theta_{q1}^2 + \theta_{q2}^2 - 2\theta_{q1}\cos(\omega_n) - 2\theta_{q2}\cos(2\omega_n) + 2\theta_{q1}\theta_{q2}\cos(\omega_n)} \quad \text{for } q = 1,2,\ldots,D$$

where $\omega_n = \exp(-2\pi j/N)$, $n = 0,1,2,\ldots,(N-1)$. We can see that spectral density is a function of the AR parameters. The non diagonal elements are zero for uncorrelated case.

In order to estimate the spectral densities we need to estimate these AR parameters. Let us estimate the parameters by writing the likelihood function in frequency domain and then maximizing it with respect to the parameters to be estimated. The computational complexity is very much reduced by working in frequency domain. The output of the first sensor is:

$$y(t) = \sum_{q=1}^{D} s_q(t) + u(t), \qquad t = 1,\ldots,N. \tag{3}$$

Taking the Fourier transform of (3) we get

$$Y(k) = \sum_{q=1}^{D} S_q(k) + U(k) \tag{4}$$

Taking the Fourier Transform of (1), for $P = 2$, and $\lambda = \exp(2\pi j/N)$

$$S_q(k) = \theta_{q1}\lambda^{-k}S_q(k) - \theta_{q2}\lambda^{-2k}S_q(k) + V_q(k)$$

$S_q(k) = V_q(k)/h_q(k)$, where $h_q(k) = (1 - \theta_{q1}\lambda^{-k} - \theta_{q2}\lambda^{-2k})$.

In (4), since S_q and U are zero mean sequences, Y is also zero mean.

$$E[S_q S_q^*] = E[V_q V_q^*]/\|h_q\|^2 \tag{5}$$

Since v. is zero mean I.I.D., $E[V_q V_q^*] = N\rho_q$ (6)

Similarly, $E[UU^*] = N\rho$. Using (5), we get $E[S_q S_q^*] = B(k) = N\left\{\sum_{q=1}^{D} \frac{\rho_q}{\|h_q(k)\|^2} + \rho\right\}$

Since the DFT {Y.} are uncorrelated, the likelihood function can be written as:

$$P[Y(1),Y(2),\ldots,Y(N) \mid \Theta] = \prod_{k=1}^{N-1} \frac{1}{\sqrt{2\pi B(k)}} \exp\left\{\frac{1}{2B(k)} Y(k)Y^*(k)\right\}$$

where $\Theta = \text{col.}[\theta_{q1}, \theta_{q2}, \rho_q, \rho]$ for all $q = 1,\ldots,D$.
The log likelihood function is

$$\ln P[\bullet] = -\frac{N}{2}\ln(2\pi) - \frac{1}{2}\sum_{k=0}^{N-1}\ln B(k) - \frac{1}{2}\sum_{k=0}^{N-1} Y(k)Y^{*}(k)/B(k)$$

we have to minimize $-\ln P(\bullet)$ with respect to Θ. Once these estimates are found the spectral density C_n becomes the known function of these parameters and the likelihood function considered in section II can be maximized with respect to the unknown DOAs.

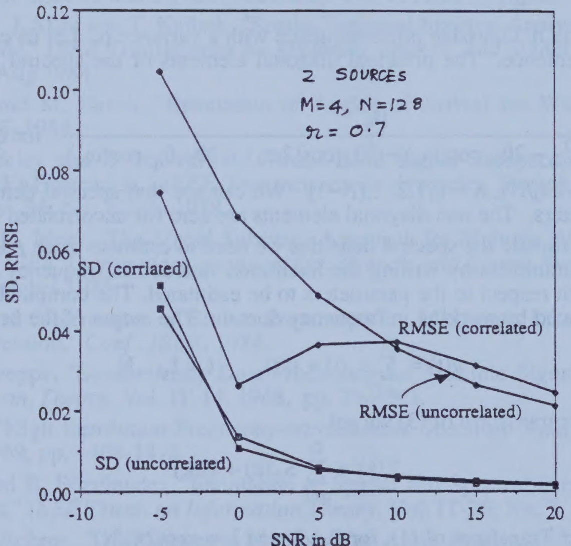


FIG.5

ARRAY PROCESSORS BASED ON m-INTERVAL POLYNOMIAL APPROXIMATION IN TIME-DEPENDENT NOISE

MOHAMMED KETEL AND LUDWIK KURZ
DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
POLYTECHNIC UNIVERSITY
BROOKLYN, NEW YORK 11201

ABSTRACT

In this paper, the theory of robust sequential array processing detectors based on m-interval polynomial approximation (MIPA) to their optimum nonlinearities is extended to include dependent sampling. A formulation is given which predicts the thresholds under N_0-dependent sampling in order to maintain the same error probabilities as in the independent sampling case. The efficiency of detection is improved if compared to the sequential MIPA array processor with independent sampling at the expense of a slight increase in structural complexity.

1. INTRODUCTION

The study of robust detectors based on the m-interval polynomial approximation was introduced by Tsai and Kurz [1,2]. These procedures were shown to be easy to implement. They require a set of quantiles [3] and related functions from the unknown noise distribution for implementation. Using an array of M sensors, the theory of m-interval approximation was extended to its multivariate counterpart [4,5,10].

With the advent of modern high-speed sampling techniques, it has become important to detect/estimate signals corrupted by unknown dependent-noise processes. For the array structure considered in [5], it may be advantageous not to demand independent samples but to sample as fast as possible. We assume the quantiles and scores are estimated with i.i.d samples, so that the nonlinearities are memoryless.* The same technique could also be used in the spatial domain when the sensors are placed close enough together so that spatial noise dependence exists. However, in this paper we will only consider temporal dependence while maintaining spatial independence of noise.

Dwyer [6], based on a scheme introduced by Woinsky and Kurz [7], extended the theory of sequential partition detectors to include q-dependent noise processes. A similar approach will be used in this paper. It is shown that knowledge of the correlation coefficient is needed in order to calculate the new thresholds necessary to maintain the same error probabilities as in the independent sampling case.

2. STRUCTURE OF THE TIME-DEPENDENT ROBUST MIPA ARRAY PROCESSOR

Let $\{\mathbf{X}^j\}_{j=1}^M$ be a sequence of independent random vectors obtained from an array of M sensors. The time vector $\mathbf{X}^j = (X_{1j}, X_{2j}, \cdots X_{ij}, \cdots X_{nj})$ is a sample from a stationary N_0-dependent random process at the j^{th} sensor. The time-space samples $X_{ij}, j = 1, 2, \cdots M$ for $i = 1, 2, \cdots n$ are real random variables representing the individual sensor output data at the i^{th} sampling instant of time. Further, $\mathbf{X} = (\mathbf{X}^1, \mathbf{X}^2, \cdots, \mathbf{X}^j \cdots \mathbf{X}^M)$ is a vector representing the total received data. Assume that the ordered vector of quantitiles $a^j = (a_{mj}^- = -\infty, a_{(m-1)j}^-, \cdots a_{1j}^-, a_{0j}^-, a_{0j}^+, a_{1j}^+, \cdots a_{mj}^+ = -\infty)$, and the parameter, γ, have been estimated, [1,5], from independent samples with p.d.f.

$$f_0(\mathbf{X}) = \gamma \prod_{i=1}^{n} \prod_{j=1}^{M} g_0(\mathbf{X}_{ij}) I_{A_0}(\mathbf{X}_{ij}) + \prod_{i=1}^{n} \prod_{j=1}^{M} h_0(\mathbf{X}_{ij}) I_{\bar{A}_0}(\mathbf{X}_{ij}) \qquad (1)$$

where $g_0(X_{ij})$ represents the nominal noise p.d.f. (usually Gaussian) and $h_0(X_{ij})$ represents the heavy tailed contaminating noise p.d.f. of unknown form. The notation $I_{A_0}(X_{ij})$ represents the indicator function of the interval A_0 which is one if $X_{ij} \in A_0$ and zero otherwise. The set $A_0 = \bigcup_{j=1}^{M} (a_{0j}^-, a_{0j}^+]$ represents the set of intervals in which the nomimal multivariate p.d.f. $g_0(\mathbf{X})$ is dominant and $\bar{A}_0$ is the complement set of $\bar{A}_0 = \bigcup_{k=1}^{m} \bigcup_{j=1}^{M} (a^{\pm}_{(k-1)j}, a^{\pm}_{kj}]$ in which the multivariate heavy tailed density, $h_0(\mathbf{X})$ is dominant [5].

---

*This estimation is performed just occasionally and with a much slower rate.

For the time-dependent robust MIPA array processor, the noise p.d.f., Eq. (1), becomes

$$f_0(\mathbf{X}) = \gamma \prod_{j=1}^{M} g_0(\mathbf{X}^j) I_{A_0}(\mathbf{X}) + \prod_{j=1}^{M} h_0(\mathbf{X}^j) I_{\bar{A}_0}(\mathbf{X}) \tag{2}$$

Let $\{X_{q+(i-1)N_0,j}\}_{i=1}^{n}$ $j = 1, 2, \cdots M, q = 1, 2, \cdots, N_0$ represent the received data with distribution function under the hypothesis $H_0, F_0(X_{1j}, X_{2j}, \cdots X_{N_{0j}})$, i.e., the data at the j^{th} sensor is a sample from a stationary N_0 dependent random process. The received data are transformed from an N_0-dimensional space into an univariate space [7] by the array which groups the beamformed output into blocks of N_0 steered data vector defined as

$$\tilde{\Lambda}_n(\mathbf{X}) = \sum_{i=1}^{n} \tilde{\Lambda}_{i,N_0} \tag{3}$$

where

$$\tilde{\Lambda}_{i,N_0} = \tilde{\Lambda}_{ip} I_{A_0}(\mathbf{X}_i) + \tilde{\Lambda}_{inp} I_{\bar{A}_0}(\mathbf{X}_i)$$

$\tilde{\Lambda}_{ip}$ represents the parametric part of the detector operating on the dependent small amplitude data. $\tilde{\Lambda}_{inp}$ is the distribution-free part of the detector operating on the dependent high amplitude data

$$\tilde{\Lambda}_{ip} = \sum_{q=1}^{N_0} \{ \sum_{j=1}^{M} L''_{qji} + \sum_{\substack{j_1=1 \\ j_1 \neq j_2}}^{M} \sum_{j_2=1}^{M} L'_{qj_1i} L'_{qj_2i} \} \tag{4}$$

where the nonlinearities

$$L'_{qji} = L'(X_{q+(i-1)N_0,j}) = [\frac{\partial}{\partial X_{qj}} g_0(X_{1j}, \cdots X_{qj} \cdots X_{N_0j})] / g_0(X_{1j}, \cdots X_{qj}, \cdots X_{N_0j}) \tag{5}$$

$$L''_{qji} = L''(X_{q+(i-1)N_0,j}) = [\frac{\partial}{\partial X_{qj}} g_0(X_{1j}, \cdots X_{qj} \cdots X_{N_0j})] / g_0(X_{1j}, \cdots X_{qj}, \cdots X_{N_0j}) \tag{6}$$

In a similar way

$$\tilde{\Lambda}_{inp} = \sum_{q=1}^{N_0}\{\sum_{j=1}^{M}\sum_{k=1}^{m} b_{jk}^{\pm} I_{A_{kj}^{\pm}}(X_{q+(i-1)N_0,j})$$

$$+ \sum_{\substack{j_1=1\\ j_1\neq j_2}}^{M}\sum_{j_2=1}^{M}\sum_{k_1=1}^{m}\sum_{k_2=1}^{m} c_{j_1k_1}^{\pm} c_{j_2k_2}^{\pm} I_{A_{k_1j_1}^{\pm}}(X_{q+(i-1)N_0,j_1}) I_{A_{k_2j_2}^{\pm}}(X_{q+(i-1)N_0,j_2})\} \quad (7)$$

where N_0 represents the number of subsamples $[X_{i+1,j}, X_{i+2,j}, \cdots X_{i+N_0,j}]$ at the j^{th} sensor used in a decision interval.

In the following we use the notation $I_{A_{kj}^{\pm}}(X_{q+(i-1)N_0,j}) = u_{jkqi}^{\pm}$

Using the received data vector **X**, the time dependent sequential array detector is constructed to test the hypotheses, H_0, H_1, with stopping boundaries, $a \approx \ell n[\frac{1-\beta}{\alpha}), b \approx \ell n[\frac{\beta}{1-\alpha}]$, where α and β are the type I and II errors, respectively [9].

Note

The random variables $\tilde{\Lambda}_{i,N_0}(i = 1, 2 \cdots n)$ can alsways be assumed independent for sufficient time delay between $\tilde{\Lambda}_{i,N_0}$ and $\tilde{\Lambda}_{i+1,N_0}$.

The basic structure of the time-dependent sequential array processor is shown in figure 1. Once the quantiles are known, the scores, $b_{jk}^{\pm}$ and $c_{jk}^{\pm}$, can be found if the functional form of the alternative is specified [5].

Let $\tilde{\Lambda}_{i,N_0}, i = 1, 2 \cdots, n$, be i.i.d. random variables and define

$$\phi(t) = E[e^{t\tilde{\Lambda}_{i,N_0}}] \quad (8)$$

as the moment generating function of $\tilde{\Lambda}_{i,N_0}$, and assume $\phi(t)$ exists and is finite for any point t, including the origin.

Lemma

Given the moment generating function in Eq. (8), a value of $t = t_0 \neq 0$ can be found so that $\phi(t_0) = 1$, where

$$\lim_{\substack{n\to\infty\\ \theta_1\to\infty}} t(\theta) = t_0 = -2\frac{E(\tilde{\Lambda}_{i,N_0})}{Var(\tilde{\Lambda}_{i,N_0})} \quad (9)$$

where θ_1 is the selected signal strength [5].

The fundamental identity for the time-dependent sequential array processing detector is given by

$$E[e^{\tilde{\Lambda}_n t}[\phi(t)]^{-n}] = 1 \tag{10}$$

for all real t for which $\phi(t)$ is finite. Where $\phi(t)$ is defined in Eq. (8).

3. PROBABILITY OF DETECTION, P(θ), AND ASND

Before obtaining the mean and variance of $\tilde{\Lambda}_{i,N_0}$ to evaluate t_0 in lemma 1, two conditions are required.

Condition 1

Assume the temporal samples at the j^{th} sensor $[X_{i+1,j}X_{i+2,j}, \cdots X_{i+N_0,j}]$ satisfy the strong mixing condition of Rosenblatt [8].

Condition 2

Assume the N_0-dimensional distribution is symmetric and the quantiles remain fixed, i.e.,

$$E(u^{\pm}_{k_1q_1}u^{\pm}_{k_2q_2}) = E(u^{\pm}_{k_2q_2}u^{\pm}_{k_1q_1}), \quad \text{for all } k_1, k_2, q_1 \text{ and } q_2 \tag{11}$$

This condition implies that the medium is homogenous. Note that relationship (11) follows from the strong mixing condition for a stationary process. However, the quantiles must remain stationary throughout a decision period. This is not a serious limitation since the quantiles can be reestimated prior to a decision period for mild non-stationarities.

To evaluate the performance based on the asymptotic relative efficiency (ARE), we shall again neglect the first term in each of $\tilde{\Lambda}_{ip}$ and $\tilde{\Lambda}_{inp}$ and assume the noise to be the same across the array.

In this case, the quantiles and the scores are independent of j. In the following, the subscript i will be dropped. For large arrays, the mean and variance of $\tilde{\Lambda}_{i,N_0}$ are reduced to

$$E(\tilde{\Lambda}_{i,N_0}|H_1) \approx 2N_0(M^2-M)\theta_1\{(E_0(L'_q)^2)^2 + (\sum_k c_k^{\pm 2} R(a_k^{\pm}))^2\} \tag{12}$$

$$Var(\tilde{\Lambda}_{i,N_0}|H_0) \approx 2N_0(M^2-M)\sum_{q_1}\sum_{q_2}\{(E_0(L'_{q_1}\cdot L'_{q_2}))^2$$

$$+\sum_{k_1}\sum_{k_2} c^{\pm}_{k_2} c^{\pm}_{k_2} E(u^{\pm}_{k_1 q_1} u^{\pm}_{k_2 q_2})\} \tag{13}$$

Where, if $\overrightarrow{X} = (X_1, \cdots X_q, \cdots X_{N_0})$ is a block vector of N_0-samples

$$E_0(L'_q)^2 = \gamma \int_{A_0}\cdots\int_{A_0}\cdots\int_{A_0}\left(\frac{\frac{\partial}{\partial X_q} g_0(\overrightarrow{X})}{g_0(\overrightarrow{X})}\right)^2 g_0(\overrightarrow{X})d\overrightarrow{X} \tag{14}$$

$$E_0(L'_{q_1}L'_{q_2}) = \gamma \int_{A_0}\cdots\int_{A_0}\cdots\int_{A_0}\left(\frac{\frac{\partial}{\partial X_{q_1}} g_0(\overrightarrow{X})}{g_0(\overrightarrow{X})}\right)\left(\frac{\frac{\partial}{\partial X_{q_2}} g_0(\overrightarrow{X})}{g_0(\overrightarrow{X})}\right) g_0(\overrightarrow{X})d\overrightarrow{X} \tag{15}$$

$$R(a_k^{\pm}) = H_0(a_k^{\pm}) - H_0(a_{k-1}^{\pm}), \tag{16a}$$

$$E[u^{\pm}_{k_1 q_1} u^{\pm}_{k_2 q_2}] = \int_{A^{\pm}_{k_1}}\int_{A^{\pm}_{k_2}} h'_0(x_{q_1}, x_{q_2}) dx_{q_1} dx_{q_2} \tag{16b}$$

Theorem

Let $t(\theta)$ represent the value of $t = t_0 \neq 0$ for which $\phi(t_0) = 1$, under independent sampling, i.e., $N_0 = 1$. Assume lemma 1 and conditions 1 and 2 hold. Then, for N_0-dependent sampling, the value of $t = t_0 = t_D \neq 0$ as $m \to \infty$ is

$$t_0 = \lim_{m\to\infty} t_0(\theta) = \frac{t(\theta)}{1 + 2\sum\limits_{v=1}^{N_0}(1 - \frac{v}{N_0})\rho^2_{\infty}(v)} \tag{17}$$

where $t(\theta) = 1 - \dfrac{2\theta}{\theta_1}$ is the solution to the fundamental identity for independent sampling. Also, $\rho_\infty(v) = \lim_{m \to \infty} \rho_m(v)$ is the asymptotic normalized correlation function, and it will be defined in the proof.

Proof

The limiting value of Eq. (13) can be found by letting the number of partitions go to infinity ($m \to \infty$), then

$$\lim_{m \to \infty} Var(\tilde{\Lambda}_{i,N_0}|H_0) = 2N_0(M^2 - M)\sum_{q_1}\sum_{q_2}(E(L'_{\infty q_1} L'_{\infty q_2}))^2 \tag{18}$$

where

$$L'_{\infty q} = \frac{\partial}{\partial X_q} f_0(\overrightarrow{X})/f_0(\overrightarrow{X}) \tag{19}$$

and $f_0(\overrightarrow{X})$ is the mixture distribution, i.e.,

$$f_0(\overrightarrow{X}) = \gamma g_0(\overrightarrow{X}) I_{A_0}(\overrightarrow{X}) + h_0(\overrightarrow{X}) I_{\bar{A}_0}(\overrightarrow{X})$$

Now let us define the asymptotic normalized correlation, $\rho_\infty(q_1, q_2)$, as follow

$$\rho_\infty(q_1, q_2) = \frac{E(L'_{\infty q_1} L'_{\infty q_2})}{E(L'_\infty)^2} \tag{20}$$

then Eq. (18) becomes

$$\lim_{m \to \infty} Var(\tilde{\Lambda}_{i,N_0}|H_0) = 2N_0(M^2 - M)(EL'^2_\infty)^2(1 + 2\sum_{v=1}^{N_0}(1 - \frac{v}{N_0})\rho^2_\infty) \tag{21}$$

By substituting equation (21) into equation (9) and for small θ, θ_1, the desired result is obtained.

From the fundamental identity Eq. (10), the probability of detection becomes

$$P(\theta) = \frac{1 - e^{bt_D(\theta)}}{e^{at_D(\theta)} - e^{bt_D(\theta)}} \tag{22}$$

Therefore, in order to guard against higher α and β due to dependence, the thresholds a and b must be adjusted by $[1 + 2\sum_{v=1}^{N_0}(1 - \frac{v}{N_0})\rho^2_\infty(v)]$.

This is an interesting result, since the thresholds would naturally be raised to guard against higher error rates if the samples were known to be dependent.

Define

$$\begin{cases} a' = [1 + 2\sum\limits_{v=1}^{N_0}(1 - \frac{v}{N_0})\rho_\infty^2(v)]a \\ \\ b' = [1 + 2\sum\limits_{v=1}^{N_0}(1 - \frac{v}{N_0})\rho_\infty^2(v)]b \end{cases} \tag{23}$$

as the new thresholds needed for a N_0-dependent process to assure the same protection against α and β as in the independent case.

Then, the asymptotic sample number for the time-dependent array processor (ASND), for the new thresholds a', b', under weak signals becomes

$$ASND = \begin{cases} \dfrac{[1 + 2\sum\limits_{v}(1 - \frac{v}{N_0})\rho_\infty^2(v)]}{E(\tilde{\Lambda}_{i,N_0})} \times \dfrac{b(e^{at(\theta)} - 1) + a(1 - e^{bt(\theta)})}{e^{at(\theta)} - e^{bt(\theta)}}, & E(\tilde{\Lambda}_{i,N_0}) \neq 0 \\ \\ \dfrac{-ab[1 + 2\sum\limits_{v}(1 - \frac{v}{N_0})\rho_\infty^2(v)]}{\sigma^2_{\tilde{\Lambda}_{i,N_0}}}, & E(\tilde{\Lambda}_{i,N_0}) = 0 \end{cases} \tag{24}$$

For efficiency calculations, let t_I and t_D represent the time between samples for the sequential array (SA) and the sequential time-dependent array (SAD) respectively. Then the efficiency of the SAD with respect to the SA is defined by,

$$\varepsilon(T_D/T_I) = \frac{ASN \cdot t_I}{ASND \cdot t_D} = \frac{ASN}{ASND_t}\frac{r_S}{N_0 + r_S - 1} \tag{25}$$

where $\frac{r_S}{\Delta'}$ is the sampling rate, Δ' is the time between independent observations and r_S is an integer. For $r_S = 1$, $\frac{1}{\Delta'}$ is the sampling rate for independent observations.

After some mathematical manipulations we obtain

$$\lim_{\substack{r_s \to \infty \\ N_0 \to \infty \\ m \to \infty}} \varepsilon(T_D/T_I) = \frac{\Delta'/2}{\int_0^{\Delta'} \rho_\infty^2(\mu)d\mu} \tag{26}$$

If $\rho_\infty^2(\mu) = e^{-\frac{k}{2\Delta'}|\mu|}$, and if $k = 3$, then $\lim_{\substack{r_s \to \infty \\ N_0 \to \infty \\ m \to \infty}} \varepsilon(T_D/T_I) = 1.5$ which is a significant improvement in efficiency for the time-dependent statistic.

4. CONCLUSIONS

In this paper we have considered the effects of the time-dependent sampling on the sequential robust MIPA array processor. In general, the MIPA array processor involves piecewise approximation of the optimum nonlinearities on the tails of the distribution. It was shown how to adjust the thresholds (a,b) in order to obtain the same error probabilities as in the independent sampling case. Since the assumption of independence is relaxed, the sampling rate increases resulting in improved efficiency over the detector with independent sampling.

REFERENCES

[1] C. Tsai and L. Kurz, "A Robustized Maximum Entropy Approach to System Identification," Proceedings of the 10th IFIP Conference, New York, (1981).

[2] C. Tsai and L. Kurz, "An Adaptive Robustizing Approach to Kalman Filtering," Automatica, Vol. 19, No. 3, 279-288, (1983).

[3] Y. Ching and L. Kurz, "Nonparametric Detectors Based on m-Interval Partitioning," IEEE Trans. Inf. Theory IT18, 241-250, (1972).

[4] R. Dwyer, "Robust Sequential Detection of Weak Signals in Undefined Noise Using Acoustical Arrays," J. Acoust. Soc. Am. 67(3), 833-841, (1980).

[5] M. Ketel and L. Kurz, "A Class of Robust Array Processing Detectors," 25th Annual Conference on Information Science and Systems, The Johns Hopkins University, (1991).

[6] R. Dwyer and L. Kurz, "Sequential Partition Detectors with Dependent Sampling," J. of Cybernetics, Vol. 10, 211-232, (1980).

[7] M. Woinsky and L. Kurz, "Nonparametric Detection Using Dependent Samples," IEEE Trans. Inf. Theory IT16, 355-358, (1970).

[8] M. Rosenblatt, "Independence and Dependence," Proc. of the Fourth Berkeley Symposium on Math. Stat. and Probl, Vol. 11, (1961).

[9] A. Wald, Sequential Analysis, New York: Wiley, (1947).

[10] M. Ketel and L. Kurz, "Array Processing Using Robust Partition Statistics," Submitted to IEEE Trans. on Signal Processing.

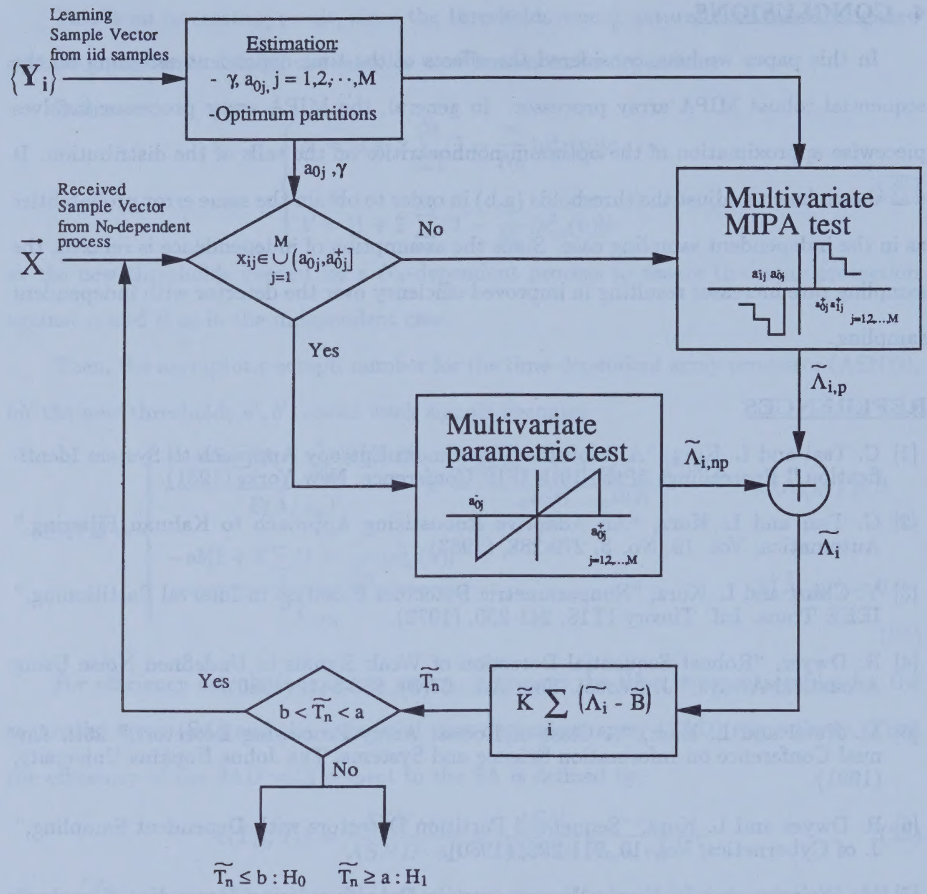

Learning
Sample Vector
from iid samples
$\{Y_i\}$
Estimation:
- γ, a_{0j}, $j = 1,2,\cdots,M$
-Optimum partitions
a_{0j} ,γ
Received
Sample Vector
from N_0-dependent
process
$\vec{X}$
$x_{ij} \in \bigcup_{j=1}^{M} (a_{0j}^-, a_{0j}^+]$
No
Yes
Multivariate
MIPA test
a_{ij} a_{0j}^-
$a_{0j}^+ a_{ij}^+$
$j=1,2,\ldots M$
$\widetilde{\Lambda}_{i,p}$
Multivariate
parametric test
a_{0j}^-
a_{0j}^+
$j=1,2,\ldots M$
$\widetilde{\Lambda}_{i,np}$
+
$\widetilde{\Lambda}_i$
$\widetilde{K} \sum_i (\widetilde{\Lambda}_i - \widetilde{B})$
$\widetilde{T}_n$
$b < \widetilde{T}_n < a$
Yes
No
$\widetilde{T}_n \leq b : H_0$
$\widetilde{T}_n \geq a : H_1$

Fig. 1: Structure of the Sequential MIPA Array for Time Dependence

A CROSS-TERM FREE JOINT TIME-FREQUENCY REPRESENTATION VIA WIGNER-VILLE DISTRIBUTION DECOMPOSITION

Joel M. Morris and Shi-E Qian
Electrical Engineering Department
University of Maryland Baltimore County
Baltimore, MD 21228-5398

SUMMARY

Over the past four decades, the Wigner and Wigner-Ville Distributions (WD) have received considerable attention in the signal processing community ([2], [8], [9]). The WD possesses many useful signal processing properties and very good resolution in the joint time-frequency domain; consequently, the application of the WD to many signal processing problems has been extensively investigated. The applications typically characterized by transient and nonstationary signals and/or time-variant systems exhibiting time-varying spectral properties: e. g., speech analysis and processing; biomedical signal analysis and medical diagnosis; remote sensing; radar and sonar; and nondestructive evaluation. The validity and usefulness of the WD as a joint time-frequency energy distribution is often challenged, however, because it has negative values and "crossterm" interference ([1], [2], [3], [4], [6], [7]). These crossterms have prevented the unqualified acceptance and anticipated performance of the WD in many signal processing applications.

In this presentation, we show that a crossterm free joint time-frequency representation -- denoted the Crossterms Deleted Representation (CDR) -- can be obtained by decomposing the WD of an arbitrary signal, via the Gabor expansion using a Gaussian synthesis function h(t), in terms of a linear combination of elementary WD's that can be easily partitioned into two summations ([5]). Thus,

$$\begin{aligned} W_s(t,f) = {} & 2 \sum_{m,n} |C_{m,n}|^2 \, exp\left[- \frac{(t-m)^2}{\sigma^2} - (2\pi\sigma)^2 \, (f-n)^2 \right] \\ & + 4 \sum_{m\neq k,\, n\neq l} \left| C_{m,n} C^{*}_{k,l} \right| \, exp\left[- \frac{(t- \frac{m+k}{2})^2}{\sigma^2} - (2\pi\sigma)^2 \, (f- \frac{n+l}{2})^2 \right] \\ & \times cos\left[2\pi((n-l)(t- \frac{m+k}{2}) + (m-k)(f- \frac{n+l}{2}))\right] \end{aligned}$$

where,

$$s(t) = \sum_{m,n} C_{m,n} h(t-m) \; exp\left[j2\pi n t \right]$$

is the Gabor expansion of the signal s(t), and

$$C_{m,n} = \int_{-\infty}^{\infty} s(t+m)\gamma^{*}(t) \; exp\left[-j2\pi n t \right] dt$$

are the Gabor coefficients, with $\gamma(t)$ biorthogonal to the synthesis function h(t). The first summation is a sum of the optimally-concentrated (by choice of h(t)) auto-WD terms, and are non-negative and entirely free of cross-terms; *this constitutes our CDR*, i.e.,

$$CDR_s(t,f) = 2 \sum_{m,n} |C_{m,n}|^2 \, exp\left[- \frac{(t-m)^2}{\sigma^2} - (2\pi\sigma)^2 \, (f-n)^2 \right]$$

The second summation is a sum of the cross-WD terms, which is an oscillating function with non-zero energy in general. Such a decomposition illustrates and isolates the mechanism of WVD negative values and cross-term interference. For the more useful case of finite-time or periodic discrete-time data, we have developed and present a Discrete Crossterms Deleted Representation (DCDR).

We believe these new results are quite useful both practically and theoretically: (1) the Gabor coefficients for this decomposition are the *only calculations required on-line*; consequently, this approach is amenable to on-line implementation; (2) these results provide new information and perspective to facilitate signal analysis and synthesis and the design of other joint time-frequency representations; and (3) the cross-terms can be easily re-inserted to obtain the original WD whenever it is needed. Moreover, these results suggest the re-examination of the WD's application to signal processing problems of interest where the "cross-terms" are the major factor in unacceptable or marginal performance and algorithm complexity. As examples, we display the CDR for several noiseless and noisy discrete-time signals of interest (see Figs.1-4), and compare them with the results for the optimally-concentrated spectrogram, the Choi-Williams Distribution, and the WD, to illustrate its potential applicability to signal detection, estimation, and classification.

REFERENCES

[1] Choi, H. I. and Williams, W. J., "Improved Time-Frequency Representation of Multi-Component Signal Using Exponential Kernals", *IEEE Trans. ASSP*, vol. ASSP-37, pp. 862-871, June 1989.

[2] Claasen, T. A. C. M. and Mecklenbrauker, W. F. G., "The Wigner Distribution -- A Tool for Time-Frequency Signal Analysis", *Philips J. Res.*, Vol. 35, 1980, pp. 217-250, 276-300, 1067-1072, .

[3] Jeong, J. and Williams, W. J., "On the Cross-Terms in Spectrograms", *Proc. Int'l. Conf. Circuits and Systems*, May 1990, pp. 1565-1568.

[4] Kadambe, S. and Boudreaux-Bartels, G. F., "A Comparison of the Existence of 'Cross-Terms' in the Wavelet Transform, Wigner Distribution, and Short Time Fourier Transform", to appear *IEEE Trans. ASSP*.

[5] Morris, J. M. and Qian, S.-E., "Wigner Distribution Decomposition and Cross-Terms Deleted Representation", submitted to *Signal Processing*.

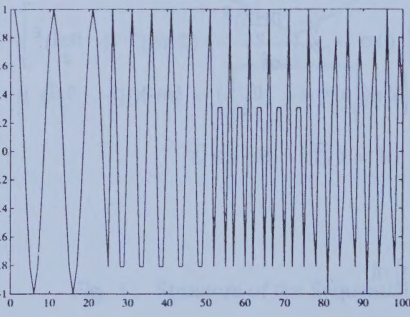
[6] Qian, S.-E. and Morris, J. M., "A Note on Eliminating Interference in Wigner Distribution of Signals", *Proc. 1989 Int'l. Symp. Comp. Arch. & DSP*, Hong Kong, 11-14 Oct 1989

[7] Rao, P. and Taylor, F. J., "Estimation of Instantaneous Frequency Using the Wigner Distributions", *Electronic Letters*, Vol. 26, No. 4, Feb. 90, pp. 246-248.

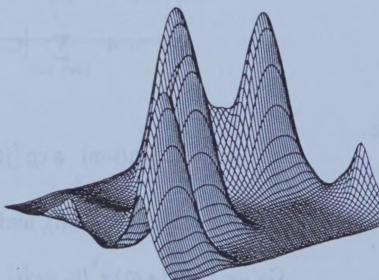
[8] Ville, J., "Thovrie et Applications de la Notion de Signal Analytique" (in French), *Càbles et Transmission*, Vol. 2, 1948, pp. 61-74.

[9] Wigner, E. P., "On the Quantum Correction for Thermodynamic Equilibrium", *Phys. Review*, Vol. 40, 1932, pp. 749.

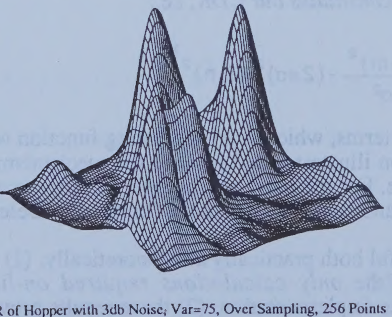
This work supported by ONR Grant #N00014-89-J1210



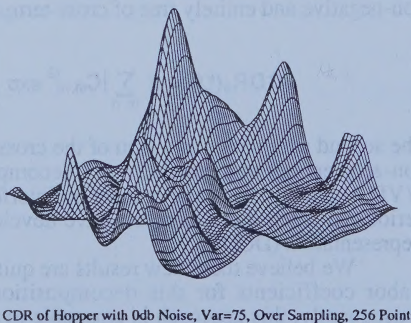
Hopper Signal, 100 Points



CDR of Hopper Signal, Var=75, Over Sampling, 256 Points



CDR of Hopper with 3db Noise, Var=75, Over Sampling, 256 Points



CDR of Hopper with 0db Noise, Var=75, Over Sampling, 256 Points

Minimum Variance Signal Estimation with Adaptive Order Statistic Filters

Peter M. Clarkson Geoffrey A. Williamson

Department of Electrical and Computer Engineering
Illinois Institute of Technology
Chicago, IL 60616

We consider the estimation of an unknown locally constant signal d embedded in zero-mean stationary noise $\{\eta_k\}$, where we have a measurement sequence $\{x_k\}$ with $x_k=d+\eta_k$, in some neighborhood of k. Several authors have approached this problem via the application of Order Statistic (OS) filters to the noisy measurements x_k [1]-[3]. An N point non-recursive OS filter with input $\{x_k\}$ produces output $y_k=\underline{a}^t\underline{x}_{(k)}$, where the components $x_{(k)}(i)$ of the vector $\underline{x}_{(k)}$ are the last N input values $x_k, \ldots, x_{k-N+1}$ ranked as $x_{(k)}(1)\leq x_{(k)}(2)\leq \cdots \leq x_{(k)}(N)$, and where $\underline{a}$ is a vector weighting these ranked values. OS filters can be defined which are optimal with respect to a much broader class of inputs than is possible for linear filters. The Minimum Variance Unbiased (MVUB) OS filter is the set of weights $\underline{a}^*$ that minimizes

$$J=E\{(y_k-d)^2\} \quad ; \textit{subject to } E\{y_k\}=d, \tag{1}$$

or equivalently minimizes the output power $E\{y_k^2\}$ subject to the same unbiasedness constraint. When the noise sequence $\{\eta_k\}$ consists of independent, identically distributed (iid) samples drawn from a symmetric density and the signal d is constant over an N point window, then it is well-known [1] that the MVUB estimate $\underline{a}^*$ is

$$\underline{a}^*=(R_{(\eta)}^{-1}\underline{1})/(\underline{1}^t R_{(\eta)}^{-1}\underline{1}). \tag{2}$$

Here, $R_{(\eta)}$ is the $(N\times N)$ autocorrelation matrix of ordered noise measurements with elements $R_{(\eta)}(i,j)=E\{\eta_{(k)}(i)\eta_{(k)}(j)\}$, and $\underline{1}^t=[1\ 1 \cdots 1]$. Practically, for locally constant d, the operator $\underline{a}^*$ must be iteratively estimated. Restrepo and Bovik [2] adopted a block-approach to the estimation of $R_{(\eta)}$ and thus of $\underline{a}^*$, while Pitas and Venetsanopoulos [3] have applied an "instantaneous gradient" based update analogous to the well-known LMS algorithm [4]. In their algorithm, however, the unbiasedness constraint is enforced by simple scaling, and one may show that the resulting algorithm does not converge to $\underline{a}^*$. Although the motivation for these algorithms has been drawn primarily from image processing, the requirement for iterative signal estimation schemes which are optimal in a MVUB sense arises frequently in spectral analysis and in array processing. For the broader class of problems that includes both dependent noise samples and noise samples drawn from an asymmetric density one may show that the MVUB OS filter is obtained from the minimization of:

$$J=E\{(\underline{a}^t\underline{\eta}_{(k)})^2\} \quad ; \textit{subject to } C^t\underline{a}=F, \tag{3}$$

where $C=[\underline{1}\ \underline{m}_{(\eta)}]$ and $F=[1\ 0]^t$, with $\underline{m}_{(\eta)}=E\{\underline{\eta}_{(k)}\}$. The unbiasedness condition is thus equivalent to the two constraints: $\underline{a}^t\underline{1}=1$ (smoothing constraint), and $\underline{a}^t\underline{m}_{(\eta)}=0$ (orthogonality constraint). The optimum filter $\underline{a}^*$ corresponding to (3) is easily obtained as

$$\underline{a}^*=R_{(\eta)}^{-1}C(C^tR_{(\eta)}^{-1}C)^{-1}F. \tag{4}$$

A block solution to (4) is possible; note however, that the matrix $R_{(\eta)}$ is symmetric

---

This work was supported in part by the National Science Foundation under grant numbers MIP-8911676 and MIP-9102620.

centrosymmetric [5], but *not* Toeplitz as is the unordered correlation matrix R_η.

We may formulate an iterative solution to (4), dubbed an Adaptive Order Statistic Filter (AOSF), with update equation:

$$\underline{a}_{k+1}=P[\underline{a}_k-\mu\underline{x}_{(k)}\,y_k]+H, \tag{5}$$

where

$$P=I-C(C^tC)^{-1}C^t \quad ; \quad H=C(C^tC)^{-1}F. \tag{6}$$

Equation (5) projects the LMS update into the space satisfying $C^t\underline{a}=F$ [6]. This AOSF filter is analogous to Frost's algorithm [6] for linearly constrained array processing and one may show that, for the class of inputs considered, the algorithm is convergent to $\underline{a}^*$ in the mean provided $0<\mu<2/\lambda_{\max}$, where $\lambda_{\max}$ denotes the largest eigenvalue of $R_{(\eta)}$.

For a general asymmetric noise density, the ordered mean $\underline{m}_{(\eta)}$ must either be available *a priori* or be estimated by data averaging. In the latter case we use an iterative estimate for $\underline{m}_{(\eta)}$ and update the constraint matrix C after each iteration. For symmetric noise densities the estimation of $\underline{m}_{(\eta)}$ can be avoided. In particular, one may show that the optimal $\underline{a}^*$ in this case satisfies $Q_N\underline{a}^*=\underline{a}^*$, where Q_N is an $(N\times N)$ matrix with $Q_N(i,j)=I_N(i,N+1-j)$ where I_N is the $(N\times N)$ identity matrix. From this the constraint matrix C and vector F can equivalently be replaced by

$$C=\begin{bmatrix} & I_{(N-1)/2} \\ \underline{1} & 0\cdots 0 \\ & -Q_{(N-1)/2}\end{bmatrix} \; ; \; N\ odd, \qquad C=\begin{bmatrix} & I_{N/2} \\ \underline{1} & \\ & -Q_{N/2}\end{bmatrix} \; ; \; N\ even,$$

$$F=[1\ 0\ \cdots\ 0]^t.$$

This modified constraint set has both performance and computational advantages: Firstly, by avoiding the estimation of $\underline{m}_{(\eta)}$ we avoid any performance degradation resulting from inaccuracies in the estimate, as well as reducing the computational requirement for the algorithm. We obtain a further considerable computational reduction by avoiding the need to recalculate C, P, and H at every iteration. Finally, by explicitly imposing the constraint $Q_N\underline{a}=\underline{a}$, we reduce the dimensionality of the problem. This produces a further performance gain by reducing the excess mean-squared error (the so-called "misadjustment") associated with the adaptive formula.

References

[1] E.H.Lloyd, "Least-squares estimation of location and scale parameters using order statistics," *Biometrika*, vol. 39, pp. 88-95, 1952.

[2] A.Restrepo and A.C.Bovik, "Adaptive trimmed mean filters for image restoration," *IEEE Trans. Acoust., Speech, and Signal Processing,* vol. ASSP-36, pp. 1326-1337, 1988.

[3] I.Pitas and A.N.Venetsanopoulos, "Adaptive filters based on order statistics," *IEEE Trans. Sig. Proc.*, vol. SP-39, pp. 518-522, 1991.

[4] B.Widrow and M.E.Hoff, Jr., "Adaptive switching circuits," *Weston Conf., IRE*, Part 4, pp. 96-104, 1960.

[5] A.Cantoni and P.Butler, "Eigenvalues and eigenvectors of symmetric centrosymmetric matrices," *Linear Algebra and its Applications*, vol. 13, pp. 275-288, 1976.

[6] O.L.Frost, III, "An algorithm for linearly constrained adaptive array processing,"*Proc. IEEE*, vol. 60, pp. 926-935, 1972.

IMAGE SEQUENCE ENHANCEMENT USING MULTISTAGE CENTER WEIGHTED MEDIAN FILTERS

Sung-Jea Ko and Thomas M. Forest
Department of Electrical and Computer Engineering
University of Michigan-Dearborn
Dearborn, MI 48128

ABSTRACT

In this paper, a filter which can be effectively used in image sequence processing, the *adaptive directional center weighted median filter* (ADCWM), is proposed. This filter utilizes a multistage filtering structure based on symmetric order statistic (SOS) operators which produce a pair of order statistics symmetric about the median. The SOS's are selected by using adaptive parameters adjusted by local image statistics. It is shown experimentally that the proposed filter can preserve image structures while attenuating noise without the use of motion estimation.

ADCWM FILTERS

Let $\{X(.,.,.)\}$ and $\{Y(.,.,.)\}$ be the input and output, respectively, of a filter. Let $\mathbf{W}$ denote the set of samples inside a $(2N+1)$ x $(2N+1)$ x $(2N+1)$ cubic window centered at (n_1, n_2, n_3), with (n_1, n_2) and (n_3), respectively, representing the spatial coodinates and the time coodinate for a spatiotemporal signal. For simplicity, we will use the vector notation $(\mathbf{n}) = (n_1, n_2, n_3)$. A CWM filter [1], [2] with window size $2L+1 = (2N+1)^3$ and central weight $2K+1$ is denoted by CWM(2L+1,2K+1), and its output is given by

$$Y(\mathbf{n}) = median\{X(L+1-K;\mathbf{W}), X(L+1+K;\mathbf{W}), X(\mathbf{n})\} \tag{1}$$

where $X(r;W)$ is the r^{th} largest sample (order statistic) among $2L+1$ samples in the window $\mathbf{W}$, and $X(\mathbf{n})$ is the central value of $\mathbf{W}$.

Statistical analysis of the CWM filter indicates that it can preserve edges under motion at the expense of noise suppression, and thus exhibits a clear tradeoff between noise suppression and motion preservation.

An adaptive version of the CWM filter has been proposed in [2]. The output $Y(\mathbf{n})$ of an ACWM filter can be expressed by (1) with a spatiotemporally varying parameter $K_{\mathbf{n}}$, defined as

$$K_{\mathbf{n}} = \begin{cases} \lceil L(1 - \frac{\sigma_n^2}{\hat{\sigma}_{\mathbf{W}}^2}) \rceil, & \text{if } \hat{\sigma}_{\mathbf{W}}^2 \geq \sigma_n^2 \\ 0, & \text{o.w.,} \end{cases} \tag{2}$$

in place of the constant K. Here $\lceil x \rceil$ represents rounding of x to the nearest integer, $\hat{\sigma}_{\mathbf{W}}^2$ denotes the sample variance of the data inside the window, and σ_n^2 represents the variance of the additive white noise which is assumed to be known. Notice that the ACWM filter varies between the median filter and the idenity filter depending on the estimated sample variance.

The noise suppression characteristic of the ACWM filter in both regions around edges and areas under motion can be further improved by incorporating spatiotemporal directional subwindows covering points spatially shifted from frame to frame by the motion. A filter of this type, the ADCWM filter, is defined as follows: Let W_i, $1 \leq i \leq M$, denote the directional subwindows of $\mathbf{W}$ with size $2L_1 + 1$, and $\hat{\sigma}_{\mathbf{W}_i}^2$ represent the sample variance of $\mathbf{W_i}$. Define the two outputs of the SOS operator for subwindow $\mathbf{W_i}$ as $Y_{Ui}(\mathbf{n}) = X(L_1 + 1 - K_{\mathbf{n_i}}; \mathbf{W_i})$ and $Y_{Li}(\mathbf{n}) = X(L_1 + 1 + K_{\mathbf{n_i}}; \mathbf{W_i})$ where $K_{\mathbf{n_i}}$ is given by (2) substituting $\mathbf{W_i}$ and L_1 for $\mathbf{W}$ and L, respectively. At the first stage of the ADCWM filter, a pair of the SOS's, $Y_{Ui}(\mathbf{n})$ and $Y_{Li}(\mathbf{n})$, are selected from subwindow $\mathbf{W_i}$ by using the adaptive parameter $K_{\mathbf{n_i}}$. The output of the ADCWM filter is then given by

$$Y(\mathbf{n}) = median\{Y_{Ui}(\mathbf{n}),\ Y_{Li}(\mathbf{n}),\ X(\mathbf{n}) \mid 1 \leq i \leq M\} \tag{3}$$

Using the ADCWM filter, a structure oriented in one direction can be preserved by a filter using a subwindow associated with that specific direction. By employing a sufficient number of subwindows, motion preservation can be achieved. This filter can utilize 1-D temporal subwindows representing the trajectory in time of a single point to preserve structures under motion. However, this may result in a reduction of noise suppression. It is possible to increase the noise suppression by using directional spatiotemporal subwindows without the loss of feature preservation properties. As an exmple, Fig. 1 shows the spatiotemporal directional subwindows (with $M = 9$ and $2L_1 + 1 = 9$) of a 3x3x3 cubic window.

EXPERIMENTAL RESULTS

An noisy image sequence, degraded by adding zero mean i.i.d. Gaussian noise of variance 100 was passed through the CWM and ADCWM filters as well as the motion-compensated temporal median (MCTM) filter [3] with $N = 1$. Fig. 2 shows the NMSE's for each of 34 frames for each of the different filters. Experimental results with actual image sequences are presented in [4]. The experimental results indicate that the ADCWM filter is an effective image sequence filter that can suppress noise while preserving image structures under motion.

REFERENCES

[1] G. R. Arce, "Multistage order statistic filters for image sequence processing,"*IEEE Trans., Acoust., Speech, and Signal Proc.*, Vol-39, pp. 1146-1163, May. 1991.

[2] S. -J. Ko, and Y. H. Lee, "Center weighted median filters and their applications to image enhancement," *IEEE Trans. Circuits and Systems*, vol. CAS-41, pp. 984-992, Sep. 1991.

[3] T. S. Huang and Y. P. Hsu, "Image sequence enhancement," *Image Sequence Analysis*, Springer-Verlag, New York, 1981.

[4] S. -J. Ko, and T. M. Forest, "Image sequence enhancement based on adaptive order statistic filters," to be submitted to *IEEE Trans. Circuits and Systems*, Oct. 1991.

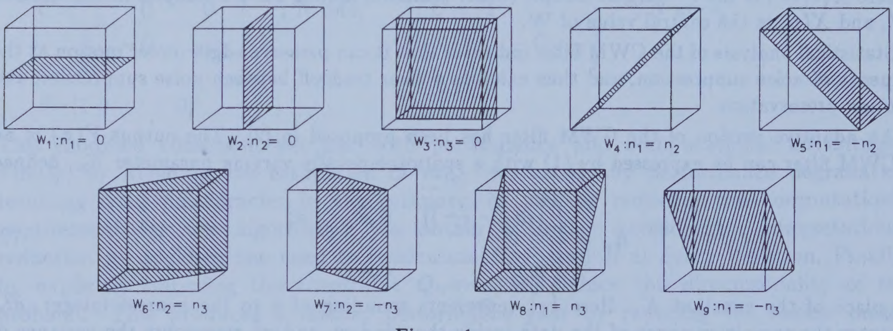


Figure 1

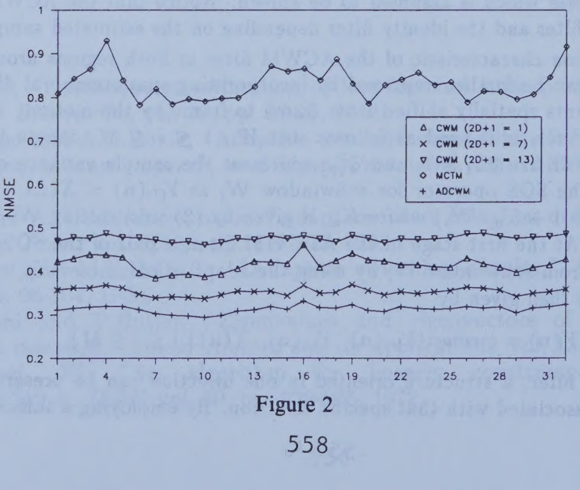


Figure 2

A Two-Stage Alpha, Beta, Gamma Estimator[†]

W.D. Blair

Code G71
Weapons Control Division
Naval Surface Warfare Center
Dahlgren, Virginia 22448-5000

I. Introduction

For tracking systems with a uniform data rate and stationary measurement noise, nonmaneuvering targets can be accurately tracked with an α, β filter. However, when the target maneuvers, the quality of the position and velocity estimates provided by the filter can degrade significantly, and for a target undergoing a large maneuver, the target track may be lost. One approach to overcoming this filtering problem is to use a small α until a maneuver is detected. Then α is increased for tracking through the maneuver [2,3]. This approach has two major problems. First, the response of the filter is significantly delayed because the gains are not increased until a maneuver is detected. Furthermore, in the α, β filter the α gain must be set artificially high to account for the absence of acceleration from the motion model. Second, after a target has returned to constant velocity from a maneuver, the decision to reduce α is often delayed because there is no bias in the higher gain α, β filter to detect.

An alternate approach which seems to be attractive is to continuously estimate the target acceleration and correct the position and velocity estimates of the α, β filter whenever a maneuver is detected as suggested in [4] for the two-stage Kalman estimator. In the two-stage $\alpha, \beta, \bar{\gamma}$ estimator, position and velocity are tracked in the first stage which is a standard α, β filter, and the acceleration is tracked in the second stage which is a standard single gain filter. The output of the acceleration filter can be used to adjust the estimates of the α, β filter when a bias is detected. This approach to tracking maneuvering target is similar to input estimation [5] with the exception that the acceleration is modeled as a stochastic process instead of a deterministic process as in input estimation.

II. Two-Stage Estimator

The "bias-free" and bias filters of the two-stage Kalman estimator in [4] corresponds to the standard constant velocity filter and standard constant state filter, respectively. The residuals of the bias-free filter are the input to the bias filter. For the filter

$$F_k = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \tag{1}$$

$$\overline{Q}_k^X = \bar{q}_k^x G_k G_k^T \tag{2}$$

$$G_k = \begin{bmatrix} \frac{T^2}{2} & T \end{bmatrix}^T \tag{3}$$

$$H_k = [1 \quad 0] \tag{4}$$

$$K_k^X = \begin{bmatrix} \alpha_k & \frac{\beta_k}{T} \end{bmatrix} \tag{5}$$

$$K_k^b = \bar{\gamma}_k S_k^{-1}, \qquad k > 0 \tag{6}$$

where $\bar{q}_k^x$ is a scalar acceleration error variance. The two-stage $\alpha, \beta, \bar{\gamma}$ estimator is given by

Prediction:

$$\overline{X}_{k|k-1} = F_{k-1} \overline{X}_{k-1|k-1} \tag{7}$$

$$A_{k|k-1} = A_{k-1|k-1} \tag{8}$$

$$U_k = F_{k-1}(I - K_{k-1}^X H_{k-1}) U_{k-1} + G_{k-1} \tag{9}$$

Smoothing:

$$\overline{X}_{k|k} = \overline{X}_{k|k-1} + \overline{K}_k^X r_k \tag{10}$$

$$A_{k|k} = (1 - \bar{\gamma}_k) A_{k|k-1} + \bar{\gamma}_k S_k^{-1} r_k \tag{11}$$

$$r_k = Y_k - H_k \overline{X}_{k|k-1} \tag{12}$$

$$S_k = H_k U_k \tag{13}$$

Output Correction:

$$X_{k|k} = \overline{X}_{k|k} + V_k A_{k|k} \tag{14}$$

$$V_k = (I - K_k^X H_k) U_k \tag{15}$$

The constant velocity filter is initialized as a standard filter and the acceleration filter is initializated with $A_{0|-1} = 0$ and $U_0 = [0 \quad 0]^T$. Eqs. (7) and (10) in steady-state are the equations for a standard α, β filter, and Eqs. (8) and (11) correspond to a single gain filter with S_k^{-1} mapping the α, β filter residuals into acceleration observations. In steady-state conditions $\alpha_k = \alpha$, $\beta_k = \beta$, $\bar{\gamma}_k = \bar{\gamma}$, and

$$U_k = U = \begin{bmatrix} \frac{T^2}{\beta} & (\frac{\alpha}{\beta} + 0.5)T \end{bmatrix}^T \tag{16}$$

$$V_k = V = \begin{bmatrix} (1 - \alpha)\frac{T^2}{\beta} & (\frac{\alpha}{\beta} - 0.5)T \end{bmatrix}^T \tag{17}$$

Since the output state error covariance is dependent on the maneuver switch which controls the acceleration correction of Eq. (14), define the switch gain

$$G_S = \begin{cases} 0, & \text{switch open} \\ \bar{\gamma}[1 + (\alpha - 0.5)\beta], & \text{switch closed.} \end{cases} \tag{18}$$

[†] This work was supported by the AEGIS Program Office, Naval Surface Warfare Center, Dahlgren Virginia.

Then the output error covariance of the filter is then given by

$$P_{k|k}(1,1) = \sigma_v^2[\alpha + (1-\alpha)G_S] \tag{19}$$

$$P_{k|k}(1,2) = \frac{\sigma_v^2}{T}[\beta + (\alpha - 0.5\beta)G_S] \tag{20}$$

$$P_{k|k}(1,3) = \frac{\sigma_v^2\beta}{T^2}G_S \tag{21}$$

$$P_{k|k}(2,2) = \frac{\sigma_v^2(\alpha - 0.5\beta)}{(1-\alpha)T^2}[\beta + (\alpha - 0.5\beta)G_S] \tag{22}$$

$$P_{k|k}(2,3) = \frac{\sigma_v^2\beta(\alpha - 0.5\beta)}{(1-\alpha)T^3}G_S \tag{23}$$

$$P_{k|k}(3,3) = \frac{\sigma_v^2\beta^2}{(1-\alpha)T^4}G_S \tag{24}$$

Since the algebraic constraint of [4] will not be satisfied by the constant acceleration motion model, the two-stage estimator with the maneuver switch closed is not identical to the standard constant acceleration filter. Thus, the position or velocity variances of the two filters can be matched. For a match between the position variances,

$$\bar{\gamma} = \frac{\hat{\alpha} - \alpha}{(1-\alpha)[1 + (\alpha - 0.5)\beta]} \tag{25}$$

where $\hat{\alpha}$ denotes the alpha gain of a standard α, β, γ filter that is desired for tracking through maneuvers.

The steady-state error covariance of the filter output resulting from the measurement errors alone when the maneuver switch is open is given by

$$V^O_{\alpha\beta\bar{\gamma}}(1,1) = \frac{\sigma_v^2}{\alpha d_1}(2\alpha^2 + \beta(2 - 3\alpha)) \tag{26}$$

$$V^O_{\alpha\beta\bar{\gamma}}(1,2) = \frac{\sigma_v^2\beta}{\alpha d_1 T}(2\alpha - \beta) \tag{27}$$

$$\begin{aligned} V^O_{\alpha\beta\bar{\gamma}}(1,3) = \frac{\sigma_v^2\bar{\gamma}\beta^2}{\alpha d_1 d_3 T^2}[\frac{2\bar{\gamma}\alpha^2}{\beta} + \alpha(2-\bar{\gamma}) + \\ \bar{\gamma}(\beta - 2) - \beta] \end{aligned} \tag{28}$$

$$V^O_{\alpha\beta\bar{\gamma}}(2,2) = \frac{2\sigma_v^2\beta^2}{\alpha d_1 T^2} \tag{29}$$

$$V^O_{\alpha\beta\bar{\gamma}}(2,3) = \frac{\sigma_v^2\bar{\gamma}\beta^2}{\alpha d_1 d_3 T^3}(\bar{\gamma}(2\alpha - \beta) + 2\beta) \tag{30}$$

$$V^O_{\alpha\beta\bar{\gamma}}(3,3) = \frac{2\sigma_v^2\bar{\gamma}^2\beta^2(\bar{\gamma}(2\alpha - \beta) + 2\beta)}{(2-\bar{\gamma})\alpha d_1 d_3 T^4} \tag{31}$$

where $d_1 = 4 - 2\alpha - \beta$ and $d_3 = (1-\alpha)\bar{\gamma}^2 + (\alpha - \beta)\bar{\gamma} + \beta$. The steady-state error variance resulting from the measurement errors alone when the maneuver switch is closed is given by

$$V^C_{\alpha\beta\bar{\gamma}} = D V^O_{\alpha\beta\bar{\gamma}} D^T \tag{33}$$

where

$$D = \begin{bmatrix} 1 & 0 & (1-\alpha)\frac{T^2}{\beta} \\ 0 & 1 & (\frac{\alpha}{\beta} - 0.5)T \\ 0 & 0 & 1 \end{bmatrix} \tag{33}$$

A gain scheduling procedure for two-stage $\alpha, \beta, \bar{\gamma}$ is given by

$$\alpha_k = \max\{\frac{2(2k+1)}{(k+1)(k+2)}, \alpha\} \tag{34}$$

$$\beta_k = \max\{\frac{6}{(k+1)(k+2)}, \beta\} \tag{35}$$

$$\bar{\gamma}_k = \max\{\frac{2(2k-1)}{k(k+1)}, \bar{\gamma}\}, \quad k > 1 \tag{36}$$

$$S_k^{-1} = \frac{1}{T^2}\min\{\frac{12}{(k+1)(k+2)}, \beta\}, \quad k > 1 \tag{37}$$

$$V_k(1) = T^2\min\{\frac{(k^2-4)(k-1)}{12k}, \frac{(1-\alpha)}{\beta}\}, k > 1 \tag{38}$$

$$V_k(2) = T\min\{\frac{(k+1)(k+2)}{12}, \frac{\alpha}{\beta} + 0.5\}, k > 1 \tag{39}$$

with $X_{0|-1} = [0 \quad 0]^T$. Note that if $Z_{0,1}/T^2$ can be extremely large relative to the dynamic range of the computer, the first three iterations of the filter should be computed analytically a *priori* to use a three point initialization.

III. Conclusions

The two-stage $\alpha, \beta, \bar{\gamma}$ estimator is simple so that it can be implemented in current systems with a modest increase in the computational burden and memory. The two-stage $\alpha, \beta, \bar{\gamma}$ estimator has the potential to provide significant performance gains in the tracking of maneuvering targets within systems that are responsible for tracking and engaging a large number of targets. Also, since the two-stage $\alpha, \beta, \bar{\gamma}$ estimator maintains an α, β filter track at all times, the command and control decisions as well as maneuver detection can be made with greater consistency with the two-stage estimator than other adaptive filtering techniques where the gains are increased. However, further research is needed for the two-stage estimator to develop improved maneuver response procedures.

References

1. Kalata, P.R."The Tracking Index: A Generalized Parameter for $\alpha-\beta$ and $\alpha-\beta-\gamma$ Target Trackers," *IEEE Trans. Aero. Elect. Syst.*, Vol AES-20, No. 2, pp. 174-182, March 1984.
2. Cantrell, B.H., *Adaptive Tracking Algorithms,* NRL Report, Washington, DC, April, 1975.
3. Blackman, S., *Multiple-Target Tracking with Radar Applications,* ARTECH House, Inc., Norwood, MA, 1986.
4. Alouani, A.T., P. Xia, T.R. Rice, and W.D. Blair, "A Two-Stage Kalman Estimator for State Estimation in the Presence of Random Bias and Tracking Maneuvering Targets," *Proc. of 30th IEEE Conf. on Dec. and Cont.*, Brighton, UK, Dec. 1991.
5. Chan, Y.T., A.G.C. Hu, and J.B. Plant "A Kalman Filter Based Tracking Scheme with Input Estimation," *IEEE Trans. Aero. Elect. Syst.*, Vol AES-15, No. 2, pp. 234-244, March 1979.

Use of Target-Oriented Process Noise in Tracking Maneuvering Targets[†]

W. D. Blair

Weapons Control Division
Naval Surface Warfare Center
Dahlgren, Virginia 22448-5000

J. D. Parker

Tactical Electronic Warfare Systems Division
Naval Surface Warfare Center
Dahlgren, Virginia 22448-5000

I. Introduction

One of the more difficult targets to track is an aircraft flying at high speeds and performing maneuvers [1,2]. In general, the maneuvers performed by aircraft flying at high speeds are constant speed turns in a plane which are known as coordinated turns. An aircraft performs a coordinated turn by controlling its lateral acceleration which is approximately orthogonal to its velocity vector [2-4]. Thus, it is logical that most of the uncertainty involved in tracking these targets is in the direction of the lateral acceleration. This information can be exploited by supplementing a standard Kalman filter with a target-oriented process noise.

Most target tracking algorithms use a Kalman filter in which the uncertainty of the target's maneuvers is modeled with a process noise [2, 5, 6]. Since the process noise is usually defined in the tracking frame, the variances for each coordinate must be set high enough to account for the largest possible maneuver in each coordinate regardless of the target's orientation. This method imposes artificially high variances for the process noise since the uncertainty due to maneuvers must be equally distributed among the three coordinates. The Kalman filter uses a time update equation to predict the error covariance matrix of the estimate at the time of the next measurement. The predicted error covariance is given by

$$P_{k+1|k} = A_k P_{k|k} A_k^T + B_k Q_k B_k^T \tag{1.1}$$

where A_k and B_k are determined by the motion model, $P_{k|k}$ is the filtered error covariance at time k, and the superscript "T" denotes the transpose of the matrix. The variable, Q_k, is the covariance matrix of the process noise and is given by

$$Q_k = \begin{bmatrix} q_k^x & 0 & 0 \\ 0 & q_k^y & 0 \\ 0 & 0 & q_k^z \end{bmatrix} \tag{1.2}$$

where q_k^x, q_k^y, and q_k^z are the variances of the process noise in Cartesian coordinates. The q_k^x, q_k^y, and q_k^z are given the same value since the direction of the target control vector is usually not observable. The covariance matrix given by (1.2) is referred to as standard process noise in this paper.

II. Target-Oriented Process Noise

A Kalman filter with target-oriented process noise is obtained by defining a process noise in the target's frame and rotating it into the tracking frame. First, the process noise covariance matrix defined in the target's frame is presented. Second, two different rotation matrices are presented. This section concludes by explaining the modifications to the standard Kalman filter which are required to obtain a filter with target-oriented process noise.

A process noise covariance matrix defined in the target's frame is given by

$$Q_k^{tg} = \begin{bmatrix} q_k^t & 0 & 0 \\ 0 & q_k^l & 0 \\ 0 & 0 & q_k^{tl} \end{bmatrix} \tag{2.1}$$

where q_k^t is the variance of the thrust acceleration, q_k^l is the variance of the lateral acceleration, and q_k^{tl} is the variance of the acceleration orthogonal to both q_k^t and q_k^l. The target's frame is defined with the X axis aligned with the target's velocity vector, the Y axis aligned with the target's lateral acceleration vector, and the Z axis aligned with the cross product of the target's velocity and lateral acceleration vectors which forms a right-handed coordinate system. The values of q_k^t, q_k^l, and q_k^{tl} are chosen to model the target's true maneuverability in the particular tracking situation. The value of q_k^t is set to λq_k^l where $0 < \lambda \leq 1$. The results from simulations have shown that for a target performing a 3 g turn $\lambda = 1$ gives the best tracking performance. For a target performing a 0.25 g turn (i.e., a slow turn), λ can be reduced to 0.25. The value of q_k^l is determined in the exact same manner that q_k^x, q_k^y, and q_k^z are determined in a Kalman filter with standard process noise. The value of q_k^{tl} is set to one of two values. For a constant velocity filter, q_k^{tl} is set to q_k^l, and for a constant acceleration filter, q_k^{tl} is set to γq_k^l where $0 < \gamma < 0.01$. The variance, q_k^{tl}, can be lowered for a constant acceleration filter tracking a coordinated turning target because there will be little acceleration uncertainty in the direction of q_k^{tl} which is orthogonal to the plane of motion. This cannot be done in the constant velocity filter because the plane defined by the velocity and lateral acceleration vectors cannot be determined without an acceleration estimate.

[†] This work was supported by the NAVSWC Independent Exploratory Development Program, Naval Surface Warfare Center, Dahlgren, Virginia.

A target-oriented process noise requires a rotation matrix to rotate the process noise defined in the target's frame into the tracking frame. Two different rotation matrices are presented. The first is suitable for a constant acceleration filter. It is determined from the target's acceleration and velocity and is given by

$$D_k = \begin{bmatrix} \frac{V_k}{|V_k|} & \frac{A^l_k}{|A^l_k|} & \frac{V_k \times A^l_k}{|V_k||A^l_k|} \end{bmatrix} \tag{2.2}$$

where $V_k = [\,\dot{x}_k \quad \dot{y}_k \quad \dot{z}_k\,]$ is the velocity vector of the target at time k, $A^l_k = [\,\ddot{x}^l_k \quad \ddot{y}^l_k \quad \ddot{z}^l_k\,]$ is the lateral acceleration vector of the target at time k, $|V_k|$ is the magnitude or length of V_k, and $V_k \times A^l_k$ is the cross product of V_k and A^l_k. The lateral acceleration vector, A^l_k, is computed by subtracting the thrust acceleration vector, A^t_k, from the acceleration vector, A_k, using the equation

$$A^l_k = A_k - \frac{(A_k \cdot V_k)}{|V_k|^2} V_k \tag{2.3}$$

where $A_k = [\,\ddot{x}_k \quad \ddot{y}_k \quad \ddot{z}_k\,]$ is the acceleration vector of the target at time k [4]. The second rotation matrix is suitable for a constant velocity filter. It is determined by the target's velocity only and is given by

$$D_k = \begin{bmatrix} \frac{\dot{x}}{S} & \frac{-\dot{y}}{Sh} & \frac{-\dot{x}\ \dot{z}}{S\ Sh} \\ \frac{\dot{y}}{S} & \frac{\dot{x}}{Sh} & \frac{-\dot{y}\ \dot{z}}{S\ Sh} \\ \frac{\dot{z}}{S} & 0 & \frac{Sh}{S} \end{bmatrix} \tag{2.4}$$

where $S = |V_k|$ and $Sh = (\dot{x}^2 + \dot{y}^2)^{\frac{1}{2}}$. Since the true state of the target cannot be determined during tracking, the filtered estimates of the target's state are used in (2.2), (2.3), and (2.4).

To obtain a Kalman filter with target-oriented process noise, the time update equation for the covariance matrix of (1.1) is modified to give

$$P_{k+1|k} = A_k P_{k|k} A^T_k + B_k D_k Q^{tg}_k D^T_k B^T_k \tag{2.5}$$

where D_k is the rotation matrix given by (2.2) or (2.4) and Q^{tg}_k is the process noise covariance matrix defined in the target's frame given in (2.1). Thus, to obtain a Kalman filter with target-oriented process noise, the standard Kalman filter is used except Q_k is replaced by $D_k Q^{tg}_k D^T_k$ in (1.1).

III. Conclusions and Future Research

Supplementing the standard Kalman filter with a target-oriented process noise allows the designer to model the real maneuverability of the target more realistically. This results in smaller filtered error variances which reduces the RMS errors. Both positive and negative correlations between the q^t_k and q^l_k components were tested, but these correlations did not improve tracking accuracy. The basic benefit of the target-oriented process noise was the reduction of measurement errors through the reduction of the modeling uncertainty in a particular coordinate.

Future research will examine the more realistic situations of single and multiple targets in the presence of clutter and false measurements by incorporating the target-oriented process noise into algorithms such as the IMMPDAF [6]. Utilizing target oriented-process noise could significantly improve tracking in a cluttered environment because the predicted position error ellipses are smaller than for filters with standard process noise. The target-oriented process noise could be used exclusively for gating while the standard process noise Kalman filter is used for filtering. This can easily be accomplished by including a separate covariance prediction equation for gating purposes such as

$$P^G_{k+1|k} = A_k P_{k|k} A_k + B_k D_k Q^{tg}_k D^T_k B^T_k \tag{3.1}$$

where $P^G_{k+1|k}$ is used only for defining the validation region. This validation region will have a smaller volume than a validation region defined by the standard covariance prediction given by (1.1) since (3.1) models the true maneuverability of the target more realistically . Thus, using (3.1) for gating will reduce the probability of false measurements being included in this region. Another subject for future research is supplementing a Kalman filter with both a target-oriented process noise and a kinematic constraint [3]. The kinematic constraint seems to be compatible with a target-oriented process noise since it does not affect the acceleration estimates in the direction orthogonal to the plane of motion.

References

1. Bar-Shalom, Y., and K. Birmiwal, "Variable Dimension Filter for Maneuvering Target Tracking," *IEEE Trans. Aero. and Elect. Sys.*, Sept. 1982, pp. 621–629.
2. Berg, R.F., "Estimation and Prediction for Maneuvering Target Trajectories," *IEEE Trans. Auto. Cont.*, March 1983, pp. 294–304.
3. Blair, W.D., G.A. Watson, and A. Alouani, "Tracking Constant Speed Target Using a Kinematic Constraint," *Proc. 23rd Southeastern Symposium on System Theory*, Columbia, SC, March 10-12, 1991.
4. Watson, G.A., and W.D. Blair, "Constant Speed Prediction Using A Three Dimensional Turning Rate," *Proc. 23rd Southeastern Symposium on System Theory*, Columbia, SC, March 10-12, 1991.
5. Blackman, S. S., *Multiple-Target Tracking With Radar Applications*, ARTECH House, Inc., Norwood MA, 1986.
6. Bar-Shalom, Y. and T. Fortmann, *Tracking and Data Association*, Academic Press, Inc., Orlando FL, 1988.
7. Houles, A. and Y. Bar-Shalom, "Multisensor Tracking of a Maneuvering Target in Clutter," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-25, No. 2, March 1989, pp. 176-189.

JUMP-DIFFUSION PROCESSES FOR OBJECT TRACKING AND DIRECTION FINDING

ANUJ SRIVASTAVA[†] MICHAEL I. MILLER[†]

ULF GRENANDER[‡]

[†] Department of Electrical Engineering
Electronic Signals and Systems Research Laboratory
Washington University
St. Louis, Missouri 63130

[‡] Division of Applied Mathematics
Brown University
Providence, Rhode Island 02912

1 Introduction

The direction of arrival problem is a well known problem in the field of array processing. In this paper we describe a new family of random sampling algorithms for use in multi-object tracking of dynamically moving targets by sensor arrays. We take a Bayesian approach where we define a posterior probabilty over the configuration space of multiple tracks and use a *jump-diffusion algorithm* to generate high probability samples from that probability measure. The mathematical foundation for *jump-diffusion* processes used for model order estimation problems of this kind is developed in [1].

The posteriori probability contains information which accomodates the two kinds of variabilities distinguishing different track scenes from one another. First, all tracks may show a large variabilty in the relative change of successive angles constituting the tracks. Second each scene is made up of a variable number and placement of the various tracks. To accomodate these variabilities we use a linear graph representation of the track which is made flexible via the introduction of two kinds of basic *transformations* on the templates. The first kind of variability in track shape is accomodated by attributing the nodes of our linear graphs with geometric rotation parameters. The transformation associated with the rotation parameters are generated using a continuous diffusion algorithm.

The second variability is accomodated by defining a rich configuration space consisting of an infinite collection of configurations of different number of tracks from which the candidate scene is selected. These configurations are sampled using the second transformation type associated with the *jump process.* The jump process operates on the entire scene adding new tracks, removing tracks, terminating tracks, fusing and splitting tracks and lengthening and shortening tracks.

Now we describe the Bayes posterior of a family of tracks.

2 Description of signal model

The model used describes the multi-track, multi-sensor scenario in the following way. Begin by considering a single signal arriving at an angle θ_l with amplitude s_l. Define the signal at the sensor array at discrete time k as $\mathbf{y}(\mathrm{k})$ made up of the superposition of the incoming signal with ambient noise and is given by

$$\mathbf{y}(k) = \mathbf{d}(\theta_l(k))s_l(k) + \mathbf{n}(k), \tag{1}$$

where $\mathbf{d}(\theta_l(k))$ is the P × 1 array of delays and gains corresponding to the P-element array and $\mathbf{n}(\mathrm{k})$ is a P × 1 0-mean complex Gaussian random vector with covariance **I**. For the cases studied here there are M signals incident upon the array, with $\{\theta_l(k), s_l(k)\}_{l=1}^{M}$, M *unknown* integer. The measured signal $\mathbf{Y}(\mathrm{k})$ in matrix form is

$$\mathbf{Y}(k) = \mathbf{D}_{\mathrm{M}}(\theta(k))\mathbf{s}_{\mathrm{M}}(k) + \mathbf{n}(k), \tag{2}$$

with matrix $\mathbf{D}_{\mathrm{M}}(\theta(k))$ a P × M matrix composed of $[\mathbf{d}(\theta_1(k))....\mathbf{d}(\theta_M(k))]$, $\mathbf{s}_{\mathrm{M}}(k)$ is the M × 1 signal vector of amplitudes at time k and $\theta(k) = [\theta_1(k), ..., \theta_M(k)]$ is the M × 1 vector of angles. Each of k = 1,2...N time samples of data are received with perhaps a different mean $\mathbf{D}_{\mathbf{M}}(\theta(k))\mathbf{s}_{\mathbf{M}}(k)$, corresponding to the dynamic motion of targets.

We assume that $\mathbf{s}_{\mathbf{M}}(k)$ is an unknown, deterministic quantity which enters into the *mean* of the observation $\mathbf{Y}(k)$ as $\mathbf{D}_{\mathbf{M}}(\theta(k))\mathbf{s}_{\mathbf{M}}(k)$, for the simplicity of presentation $\mathbf{s}_{\mathbf{M}}$ is assumed to be known. The likelihood becomes

$$L(data|\theta) \propto \prod_{k=1}^{N} e^{-\left|\mathbf{Y}(k)-\mathbf{D}_{\mathbf{M}}(\theta)\mathbf{s}_{\mathbf{M}}(k)\right|^2}. \tag{3}$$

Note, this is the standard narrowband signal model developed in [2] with identical likelihood expression as used in [3].

3 Multiple-track configuration space and Baye posterior

Now define the configuration space of which the M-track scenes are elements. A track is completely defined by the sequence of L angles encoding that track according to

$$c_1 = \{\theta(k)\}_{k=1}^{L} \in [0, 2\pi)^L. \tag{4}$$

Let there be M tracks present in the scene, M unknown. Assuming that all the tracks are of length L (this can be done by setting the amplitude s to zero during all data measurements intervals for which there is no source component), an M-track scene is $c_M = (c(1), c(2)....., c(M)) \in [0, 2\pi)^{ML}$, and the full configuration space becomes

$$\mathcal{C} = \bigcup_{M=0}^{\infty} [0, 2\pi)^{ML}. \tag{5}$$

$\mathcal{C}$ is a countable, infinite union of subspaces each having an associated value of M. This fact highlights the important issue of the unknown model order estimation problem. That is, we do not know the number of tracks beforehand. Each scene of M objects has a variable number ML of parameters, with M a parameter to be found by the algorithm.

The Bayes posterior $\pi(c_M, M|data)$ for the M-track scene can be constructed as follows. Define the Von-Mises prior $\pi(c_1)$ for a single track

$c_1 = \{\theta(k)\}_{k=1}^{L}$ to have Gibb's potential

$$log(\pi(c_1)) \propto a \sum_{k=2}^{L} cos(\theta(k) - \theta(k-1)), \tag{6}$$

with the M-track scene prior having potential

$$log(\pi(c_M, M)) \propto \sum_{l=2}^{M} \sum_{k=2}^{L} a(cos(\theta_l(k) - \theta_l(k-1))) + E(M), \tag{7}$$

where E(M) represents the prior on the number M of tracks in the scene.Notice, for a a large positive number the prior favors smooth tracks. Then the posterior becomes

$$log(\pi(c_M, M|data)) \propto \sum_{l=1}^{M} \sum_{k=2}^{L} a(cos(\theta_l(k) - \theta_l(k-1))) + E(M) \tag{8}$$

$$- \sum_{k=1}^{L} |\mathbf{Y}(k) - \mathbf{D_M}(\theta(k))\mathbf{s_M}(k)|^2 .$$

4 Sampling the multi-track space

Here we define the algorithm for sampling the multi-track space $\mathcal{C}$. First we construct a Markov *jump-diffusion* process $C(t) \in \mathcal{C} = \sum_{M=0}^{\infty}[0, 2\pi)^{ML}$ which visits the candidate solutions according to the posterior density

$$\pi(c_M, M|data) = Z^{-1} e^{-E(c_M, M|data)}, \tag{9}$$

with the potential

$$E(c_M, M|data) = \sum_{l=1}^{M} \sum_{k=2}^{L} a(cos(\theta_l(k) - \theta_l(k-1))) + E(M) \tag{10}$$

$$- \sum_{k=1}^{L} |\mathbf{Y}(k) - \mathbf{D_M}(\theta(k))\mathbf{s_M}(k)|^2 .$$

The jump process part of the random sampling algorithm deals with the assignment of tracks and the choice of model order in two ways. First, the individual tracks are developed by probablistically placing the track-segments sequentially in the associated track configuration. Secondly, the jump process moves among the subspaces of variable numbers of tracks via the addition and deletion of tracks. The jump moves at both the levels are performed simultaneously.

For hypothesizing the existence of new tracks and the disappearance of a faulty track hypothesis as well as growing and shrinking tracks we use classical ideas associated with birth-death processes, where a birth corresponds to a newly hypothesized track in the scene and a death the removal of one. These births/deaths are two of a family of *simple jump moves*,others corresponding to splitting and fusing of tracks, with the simple moves transforming one model to another. The jump intensities of simple moves are governed by the posterior potential, with the process visiting configurations of higher probabilties for longer times.

While the algorithm is jumping from one sub-space to another sub-space at random times the process dynamics within a subspace are governed by the diffusion process . The diffusion process satisfies a family of stochastic differential equations, each stochastic differential equation associated with a different subspace of dimension appropriate for the number of tracks in that sub-space. For example, for the sub-space containing M tracks the diffusion follows Langevin's stochastic differential equation

$$dC(t) = \nabla_M E(C(t))dt + \sqrt{2}dW_M(t), \tag{11}$$

where $W_M(t)$ is a standard vector Wiener process, $E(C(t))$ is the potential of the posterior and dC(t),$\nabla_M E(C(t))$ and $dW_M(t)$ are all of dimension ML × 1.

Now we rely on results from [1] which are presented in the form of a proposition.

Proposition : If the jump-diffusion process has the properties that

1. within any fixed sub-space $[0, 2\pi)^{ML}$ the diffusion satisfies the stochastic differential equation

$$dC(t) = \nabla_M E(C(t))dt + \sqrt{2}dW_M(t) \tag{12}$$

2. the jump intensities satisfy Theorem 4.4 and its corollary 1 of [1]

then,

C(t) converges in density to $\pi(c_M, M)$ over the configuration space $\sum_{M=0}^{\infty}[0, 2\pi)^{ML}$.

For details of the statement and the proof refer to [1].

5 Simulation results

To illustrate the use of this algorithm it was run on the simulated data sets with track length L=10 and sensor having 10 elements at half wavelength spacing. Shown in figure 1 are the tracks which have been estimated using the algorithm. The 2-D contour plots of power as a function of spatial angle indicate the positions of possible sources.The power measure was generated using the minimum variance distortionless response (MVDR) beamforming [4]. Superimposed via the dashed line is the "true track", with the solid line the estimated track generated using the jump-diffusion algorithm. The algorithm was run with the jump part of the process adding angles to the path, and the diffusion part constantly adjusting the overall path. The top plot shows high SNR data with the associated power distribution peaks closely following the true track. The bottom plot shows low SNR data where the power peaks are displaced from the true track. Notice, the track generation using the jump-diffusion process supports the accurate estimation of track.

References

[1] U. Grenander and M. I. Miller. Jump-diffusion processes for abduction and recognition of biological shapes. *Monograph of the Electronic Signals and Systems Research Laboratory*, 1991.

[2] R. Schmidt. *A signal subspace approach to multiple emitter location and spectral estimation.* Ph.D. Dissertation of Stanford University, Palo Alto, CA., Nov. 1981.

[3] M.I. Miller and D. R. Fuhrmann. Maximum likelihood narrow-band direction finding and the em algorithm. *IEEE Acoust. Speech and Signal Processing*, 38, No.9(38, No.9):560–577, 1990.

[4] N. Owsley. Sonar array processing. In Simon Haykin, editor, *Array Signal Processing*. Prentice-Hall, New Jersey, 1985.

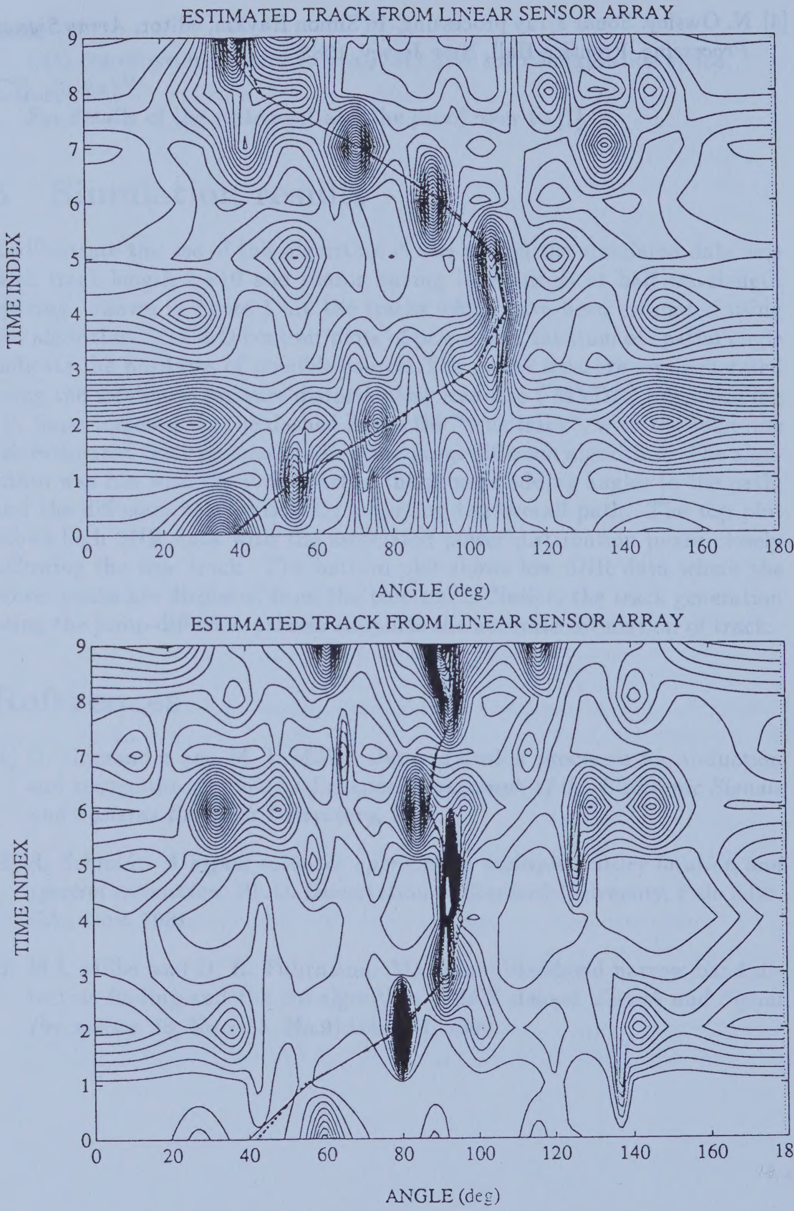


Figure 1: 2-D contour plots of the distribution of energy as a function of spatial angle (x-axis) and time (y-axis). Superimposed via dashed lines is the "true track trajectory" and superimposed via the solid line is the estimated track from the jump-diffusion algorithm. The top plot shows high SNR sensing data for which the power peaks are close to true track; bottom plot shows the low SNR where the power peaks are displaced from the truth but the higher level algorithm with priors allows for accurate track generation.

Bounds on Maximum Delay in Networks with Deflection Routing

Jack Brassil† *and Rene Cruz*‡

Dept. of Electrical & Computer Engineering
University of California, San Diego
La Jolla, CA 92093-0407

ABSTRACT

We consider the problem of finding the worst case packet transit delay in networks using deflection routing. We study several classes of networks, including many topologies for which deflection routing has been proposed or implemented (e.g., hypercube, Manhattan Street Network, shuffle-exchange network). We derive new upper bounds on the evacuation time of batch admissions, and present simple proofs for several existing bounds. We also derive bounds on worst case transit delay for certain networks admitting packets continuously. The results have application in both protocol design and the determination of the required capacity of packet resequencing buffers.

1. Introduction

Consider an ideal communication network with corresponding directed, strongly connected graph $G = (V,E)$ having $|V|$ vertices (nodes) and $|E|$ edges (directed communication links). We assume the network is packet switched and time slotted, with unit slot time. Suppose the network uses *deflection* routing [1]. In deflection routing nodes attempt to forward each arriving packet along a shortest hop path to its destination. At most one packet can be transmitted on each link in a time slot. Each packet arriving to a node other than its destination is transmitted in the next time slot; no packets are queued for forwarding in future time slots. Contention for access to a communication link may arise, causing some packets to be misdirected or *deflected,* forcing the deflected packets to travel longer paths to their destinations. To ensure deflection routing operates without packet loss, we restrict our attention to networks of nodes whose in-degree is no larger than out-degree.

In this paper we develop upper bounds on the worst case packet delay in several classes of networks using deflection routing. Unless specifically stated, we make no assumptions regarding the distribution of destinations of admitted packets. We examine 2 distinct types of offered load. We first study the evacuation time of packets admitted in a batch in a single time slot *(batch admissions),* with no subsequent admissions. The evacuation time is the number of time slots taken by the last packet in the network to reach its destination. We also study the maximum transit delay of packets admitted over multiple time slots *(continuous admissions).* Here we take the transit delay to be the number of hops required for an *admitted* packet to reach its destination.

1.1. "Go-back" Networks

We begin by defining classes of networks in terms of the structure of their corresponding graphs. Consider a graph G with three distinct nodes a, b, $c \in V$ with edge $(a,b) \in E$. Let d_{xy} denote the shortest path distance (in hops) between node x and node y. Suppose that node b is not along a shortest path from node a to node c (i.e., $d_{ac} \le d_{bc}$). Thus, the length of a shortest path *abc* (i.e., from node *a* to node *c* via node *b*) is $1 + d_{bc} - d_{ac}$ hops longer than the shortest path *ac*.

---

† On leave from AT&T Bell Laboratories, Middletown, NJ. ‡ This work was partially supported under grant NSF NCR-8904029.

Now consider the paths taken by packets traversing the network corresponding to *G*. We say a packet is *deflected* if in a time slot it does not travel to a node which is closer to its destination. We define the *distance* of a deflection as the difference in a deflected packet's destination distance (in hops) immediately before and after a deflection. A deflection from node *a* into node *b* (w.r.t. destination node *c*) has distance $d_{bc} - d_{ac}$. We define the deflection *index* (D_I) of the network (or graph) as

$$D_I \triangleq 1 + \max_{a,b,c \in V} d_{bc} - d_{ac} , \qquad (a,b) \in E. \tag{1.1}$$

The index equals the maximum number of additional hops a deflection can add to a packet's path length.

We next define classes of networks in terms of the deflection distances which can be realized by traversing packets. A "go-back-d" network is characterized by a *constant* deflection distance *d* for any realizable deflection. Hence, $D_I = d + 1$. Examples of "go-back-1" networks include the binary n-cube and n-dimensional mesh ($n \geq 2$). A Manhattan Street graph with an integer multiple of 4 nodes in each row and column is a "go-back-3" network [2]. One can easily construct a chordal ring of degree 2 to form a "go-back-d" network for arbitrary values of $d \geq 1$.

A "go-back-all" network with diameter *L* is defined by the following property: a deflection of a packet at destination distance *i* (i.e., exactly *i* hops from its destination) has deflection distance $L - i$. That is, a deflected packet finds itself at the maximum possible distance from its destination in the time slot immediately following the deflection. An example of a network that has been studied as a "go-back-all" network is the folded shuffle-exchange [3]. Finally, we define a general "go-back" network as one where deflection distance depends on the particular choice of vertices $a, b, c \in V$. Examples of general "go-back" networks include many graphs of "irregular" topology.

Hajek [4] has derived the following three upper bounds on the evacuation time of batch admissions in certain networks operating with deflection routing. Suppose a batch of *p* packets is admitted to a network with diameter *L* and deflection index D_I. Then if all packets share a *single destination,* the evacuation time (T) satisfies

$$T \leq L + D_I(p-1) \qquad \textit{single destination.} \tag{H1}$$

Next suppose that arriving packets are assigned to links by a *"one-pass" priority* rule, described as follows. At each node in every time slot, order arriving packets by nondecreasing destination distance. Then, starting with the packets closest to their destinations, assign each packet to an outgoing link along a shortest path, if available. If such a link is unavailable, assign the packet to a randomly chosen available outgoing link. Under the one-pass priority rule, evacuation time satisfies

$$T \leq L\,p \qquad \textit{arbitrary topology, multiple destinations,} \tag{H2}$$

$$T \leq n + 2(p-1) \qquad \textit{binary n-cube, multiple destinations.} \tag{H3}$$

In the remainder of this paper, we show how the above evacuation bounds can be tightened or more simply proved, and consider other related problems. In particular we study the maximum transit delay of packets admitted continuously. We know of no published results bounding worst case delay under continuous admissions. However it is known that *livelock* (i.e., the continuous circulation of admitted packets) can be avoided in routing schemes by the introduction of a priority mechanism [5].

2. Analysis

Let i denote a packet's remaining distance to destination. A packet reaching its destination is taken to be at distance $i = 0$, and is immediately removed from the network. Let time slot t refer to the interval $[t-1, t)$. Define the following variables:

$N^t(i) \triangleq$ *total number of packets at distance i at the end of time slot t,*

$N^t \triangleq$ *total number of packets in network at the end of time slot* $t = \sum_{i=1}^{L} N^t(i)$,

$D^t(i, l) \triangleq$ *total number of packets at distance i deflected to distance l in time slot t,*

$D^t(i) \triangleq$ *total number of packets at distance i deflected in time slot* $t = \sum_{l=i}^{L} D^t(i, l)$,

$D^t \triangleq$ *total number of deflections in network in time slot* $t = \sum_{i=1}^{L} D^t(i)$.

The evolution of packet destination distances in a "go-back-d" network with diameter L is described by the following system (or *evolution*) equations:

$$N^{t+1}(i) = \begin{cases} N^t(i+1) - D^t(i+1) & i = 0, 1, \ldots, d \\ N^t(i+1) - D^t(i+1) + D^t(i-d+1) & i = d+1, \ldots, L-d \\ N^t(i+1) + D^t(i-d+1) & i = L-d+1, \ldots, L-1 \\ D^t(i-d+1) & i = L. \end{cases} \tag{2.1a}$$

The system equations for a "go-back-all" network are

$$N^{t+1}(i) = \begin{cases} N^t(i+1) - D^t(i+1) & i = 0, 1, \ldots, L-1 \\ \sum_{j=1}^{L} D^t(j) & i = L, \end{cases} \tag{2.1b}$$

while a general "go-back" network satisfies

$$N^{t+1}(i) = \begin{cases} N^t(i+1) - D^t(i+1) + \sum_{j=(i-D_I+1)^+}^{i} D^t(j,i) & i = 0, 1, \ldots, L-1 \\ \sum_{j=L-D_I+1}^{L} D^t(j,L) & i = L. \end{cases} \tag{2.1c}$$

We begin with a useful result bounding the evacuation time for a batch of N^0 packets admitted to the network immediately prior to the beginning of the first time slot (i.e., $t = 0$).

Lemma 2.1: Suppose a batch of N^0 packets is admitted to a "go-back" network with diameter L and deflection index D_I. If

$$N^t(i+1) \geq 1 \quad \Rightarrow \quad N^{t+1}(i) \geq 1 \qquad i = 0, 1, \ldots, L-1, \tag{2.2}$$

then the network evacuates in time

$$T \leq L + D_I\,(N^0 - 1). \tag{2.3}$$

Discussion: Note that we have not specified a rule for assigning arriving packets to outgoing links. Equation 2.3 is identical to (H1), but is not restricted to packets sent to a single destination. The condition (2.2) states that at least 1 packet among those found at *every* destination distance i must move 1 hop closer to its destination in each time slot. This guarantees that at least 1 packet found at distance i at time t will reach its destination at time $t+i$, i.e.,

$$N^t(i) \geq 1 \quad \Rightarrow \quad N^{t+i}(0) \geq 1. \tag{2.4}$$

Proof: Following the proof of (H1) [4], we first show that at least 1 packet reaches its destination in the interval $t \in [1, L]$. We then show that for each time $t_j = L + jD_I$, $j = 0,1,\ldots$, at least 1 packet reaches its destination in $[t_j + 1,\ t_j + D_I]$, until the network evacuates.

Let $Q \triangleq \{ i : N^0(i) \geq 1 \}$. By assumption the network is not empty initially (i.e., $|Q| \geq 1$). Eq. 2.4 indicates that for $q \in Q$ we have $N^0(q) \geq 1 \;\Rightarrow\; N^q(0) \geq 1$. Hence, at least one packet reaches its destination in the interval $[1, L]$.

Prior to beginning the second part of the proof, we introduce several definitions. Let $i^t_{\max}$ represent the destination distance of the packet furthest from its destination at time t, i.e. $i^t_{\max} \triangleq \max\ \{ i : N^t(i) \geq 1\}$. We may then define the evacuation time as $T \triangleq \min\ \{ t : i^t_{\max} = 0\}$. Finally, we say that a *type A* deflection occurred in time slot t iff $i^{t+1}_{\max} \geq i^t_{\max}$. That is, in the time slot immediately following a *type A* deflection, at least one packet is as far from its destination as was the furthest packet in the previous time slot. Note that a packet deflected in a *type A* deflection at time t must have been at some destination distance $i \in [\ (i^t_{\max} - D_I)^+ + 1,\ i^t_{\max}\ ]$ at time t. Note that if no *type A* deflection occurs at time t, then $i^{t+1}_{\max} = i^t_{\max} - 1$. Hence only *type A* deflections can postpone a network's evacuation.

We next present the second part of the proof; if the network is not evacuated at a time $t_j = L + jD_I$, $j = 0,1,\ldots$, then at least 1 packet reaches its destination in $[t_j + 1,\ t_j + D_I]$. Consider such a fixed time t_j. Since the network is not empty by assumption, a *type A* deflection must have occurred at some time prior to t_j. Let t_d be the time slot of the first *type A* deflection which prevents the network from evacuating prior to time t_j, i.e. $t_d \triangleq \min\ \{t : i^{t+1}_{\max} \geq t_j - t\}$. At time $t_d + 1$, at least 1 packet is sufficiently far from its destination that it can not be reached by time t_j, i.e. $i^{t_d+1}_{\max} \geq t_j - t_d$. Moreover, we have $i^{t_d}_{\max} \leq t_j - t_d$. Since $D_I - 1$ is the maximum increase in a deflected packet's destination distance, we have

$$t_j - t_d \leq i^{t_d+1}_{\max} \leq i^{t_d}_{\max} + D_I - 1 \leq t_j - t_d + D_I - 1. \tag{2.5}$$

By (2.4) at least 1 packet at distance $i^{t_d+1}_{\max}$ at time $t_d + 1$ must reach its destination at time $t_d + 1 + i^{t_d+1}_{\max}$. So (2.5) indicates that at least 1 packet must reach its destination in the interval $[t_j + 1,\ t_j + D_I]$. □

3. Admissions to a Single Destination

We first consider the evacuation time of a batch of packets all destined for the same node. With a single destination demand, all packets visiting a node are equidistant from the destination node. Consequently, all contending packets have identical destination distance. In the following Proposition, we state that the single destination demand itself is sufficient to satisfy condition (2.2), so Lemma 2.1 may be used to bound network evacuation time. Note that this result is identical to (H1).

Proposition 3.1: Suppose a batch of N^0 packets is admitted to a network with diameter L and deflection index D_I. If all packets are destined for the same node, then the evacuation time satisfies

$$T \leq L + D_I(N^0 - 1). \tag{3.1}$$

Proof: In any time slot prior to T, select any node (other than the destination) holding at least one packet. All packets at the chosen node are equidistant from the destination node. At least one outgoing edge from the chosen node must be incident upon a node exactly one hop closer to the destination node d. Hence, at any nonempty node at least 1 packet must move one hop closer to the destination. It follows that at least 1 packet among those at every destination distance moves 1 hop closer to the destination in each time slot. Hence, condition (2.2) is satisfied, allowing us to apply Lemma 2.1, and (3.1) follows. □

3.1. Batch and Continuous Admissions in "Go-back-d" Networks

We next examine the special case of admissions to a single destination "go-back-d" network. We will show that the batch evacuation time bound (H1) can be reduced by a factor of $d+1$, approximately. Further, we will show that this bound applies to certain cases of continuous admissions as well.

Suppose packets are admitted continuously, and let $i(t)$ denote a marked packet's remaining destination distance at time *t*. Define the *admission class* of a packet admitted at time t_0 with initial distance $i(t_0)$ as

$$\{ t_0 + i(t_0) \} \bmod D_l \in \{0, ..., D_l - 1\}. \tag{3.2}$$

There are exactly $D_l = d+1$ admission classes. Recall that each deflection in the "go-back-d" network has distance *d*. A packet admitted at time t_0 at distance $i(t_0)$ can only reach the destination at some time $t = t_0 + i(t_0) + jD_l,\ j = 0, 1, ...$.

Fact: If all packets admitted to a "go-back-d" network are sent to a single destination, then any contending packets belong to the same admission class.

The significance of the above fact is that for a single destination demand with either batch or continuous admissions, we may apply the result of Proposition 3.1 to to each admission class separately.

We now introduce a model for a single destination demand in a "go-back-d" network. Consider a regular network of degree *k*. Suppose that each node has a 1 packet capacity buffer for each incident link (i.e., each node can hold *k* packets). Note that the buffers at all nodes (other than the destination) may be viewed collectively as a "distributed queue" with combined capacity of $k(|V|-1)$ packets. We may logically partition this "queue" into D_l parallel queues of packets, one corresponding to each admission class. The capacity of the l^{th} queue is

$$C_l \triangleq k \sum_{j: l+1+jD_l \le L} R_{l+1+jD_l} \qquad l = 0, ..., D_l - 1, \tag{3.3}$$

where $R_i \triangleq$ *total number of nodes at destination distance i.*

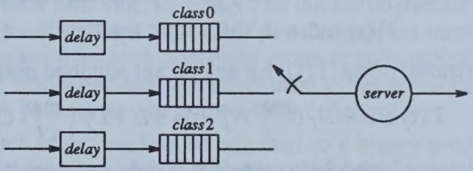


Figure 1: The "parallel queue" model for a regular "go-back-2" network of degree *k*. Newly admitted packets suffer a variable delay equal to their initial destination distance, and are then admitted to the queue corresponding to their admission class. At least one packet from each nonempty queue is serviced (i.e., reaches the destination) every D_l time slots.

In our model a packet admitted to the network at time t_0 with destination distance $i(t_0)$ is initially delayed for $i(t_0)$ time slots before being admitted to the "queue" for its admission class. The D_l queues are served in round robin fashion by a "server," with service corresponding to absorption at the destination node. At least 1 packet in each nonempty queue is serviced every D_l time slots. A deflection represents a reordering of service to packets *within a single queue*. Thus, the bound on worst case delay is determined by the distribution of packets among the queues. We summarize our result for batch demands in the following proposition.

Proposition 3.2: Suppose a batch of N^0 packets is admitted to a "go-back-d" network, with all packets destined for a single node. Let N_j^0 be the number of class *j* packets initially admitted. The evacuation time of class *j* packets (T_j) satisfies

$$T_j \le L + D_I\,(N_j^0 - 1), \qquad j = 0,\ldots,D_I - 1. \tag{3.4}$$

The maximum evacuation time of any admitted packet is

$$T \le L + D_I\,(\max_j N_j^0 - 1) \le L + D_I\,(\max_j C_j - 1). \tag{3.5}$$

Discussion: In many regular networks for which deflection routing has been proposed [6, 7] each of the D_I "queues" in our model has approximately the same capacity. Proposition 3.2 indicates that the evacuation time for fully loaded single destination "go-back-d" networks is approximately $D_I = d + 1$ times smaller than (H1) or (3.1). For example, the binary n-cube can admit a batch of $N^0 \le n(2^n - 1)$ packets destined to a single node. This batch may be partitioned among the 2 "queues" having capacities $C_0 = n\,2^{n-1}$ and $C_1 = n(2^{n-1} - 1)$ packets. It follows that the evacuation time of a fully loaded binary n-cube satisfies $T \le n - 2 + n2^n$, which is approximately half of (H1).

The "parallel queue" model for "go-back-d" networks enables us to extend our results to include *continuous* admissions to a single destination. However, we must ensure that our choice of packet-to-link assignment rule avoids livelock [8]. We now introduce a *longest-in-network* priority assignment rule, described as follows. At each node, arrange packets in nondecreasing order by the time each has spent in the network. Beginning with the packet(s) longest in the network, assign each packet to a preferred outgoing link (i.e., along a shortest path) if available. If unavailable, assign the packet to a randomly chosen available outgoing link.

In terms of our network model, a "reordering" of service to packets due to a deflection does not permit "younger" packets to take precedence over "older" packets. Note, however, that it is possible for "younger" packets to reach the destination before older packets. Again we use the fact that at least one packet in each nonempty queue is serviced (i.e., reaches the destination) every D_I time slots. We summarize our result in the following proposition.

Proposition 3.3: Suppose packets are admitted continuously and destined for a single node in a "go-back-d" network operating with the *longest-in-network* assignment rule. Let N^τ (N_j^τ) denote the number of packets (of class j) in the network at the end of time slot τ. The maximum transit delay ($T_j(\tau)$) for a class j packet admitted during time slot τ satisfies

$$T_j(\tau) \le L + D_I\,(N_j^\tau - 1) \le L + D_I\,(C_j - 1), \tag{3.6}$$

and the maximum transit delay ($T(\tau)$) for any packet admitted during time slot τ satisfies

$$T(\tau) \le L + D_I\,(\max_j N_j^\tau - 1) \le L + D_I\,(\max_j C_j - 1). \tag{3.7}$$

The "parallel queue" model also provides insight into packet delay in general "go-back" networks. The following discussion may apply to either batch or continuous admissions. Suppose admitted packets are destined for a single node, and the deflection distances suffered by packets are not constant. In terms of our model, a deflected packet may now possibly move from one "queue" to another (i.e., a deflection can effectively change the "admission class" of a packet). If the deflections distances suffered by packets are varied, there will be a tendency for packets to distribute evenly among the queues (i.e., "random" jockeying). Indeed, it appears that this "randomization" can reduce delay if certain queues are initially heavily loaded while others are relatively empty. In the next section we consider one such situation.

3.2. Batch Admissions in the "Go-back-all" Network

Consider a batch of N^0 packets admitted to a "go-back-all" network and destined to a single node. Proposition 3.1 indicates that the network evacuates in time

$$T \le L + L\,(N^0 - 1). \tag{3.8}$$

We now argue that the evacuation time is "likely" to be considerably less. As we will show, a shorter evacuation time is due to randomization introduced by varied packet deflection distances.

In the time slot immediately following a deflection, at least 1 packet is found at destination distance $i = L$. Hence, (2.4) indicates that at least one packet must exit L time slots later. In summary, a deflection at time t guarantees that a packet exits at time $t + L + 1$. Now suppose that in each time slot where there are 2 or more packets in the network the following 2 conditions are met: **a)** at least 1 packet at each destination distance moves one hop closer to its destination, and **b)** some other packet is deflected. Satisfying these conditions would cause the network to evacuate in time $T \leq L + (N^0 - 1)$, approximately L times faster than (3.8). Paradoxically, very long delays might be incurred by having too *few* deflections. This suggests that worst case evacuation time can be reduced by implementing a policy where some packets are voluntarily deflected.

Unfortunately, it does not seem possible to guarantee that both conditions can be met in a distributed fashion. However, in time slots where the network is heavily occupied ($N^t \gg 0$), both conditions are likely to be met, assuming the use of a conventional packet-to-link assignment rule. Further, in time slots where the network is lightly loaded, deflections should be relatively few in number, and packets should quickly reach the destination.

4. Admissions to Multiple Destinations

We next study the evacuation time of a batch of packets, each of which may have a different destination. Note that under a multiple destination demand, packets located at the same node in a time slot can be at different destination distances.

4.1. Batch Admissions in the Hypercube

In this section we show that the evacuation time for a batch of N^0 packets admitted to a binary n-cube is bounded by $n + 2(N^0 - 1)$. This result is identical to (H3), but we use a different assignment rule, and the proof is considerably simpler.

Call the following distributed assignment rule the *guaranteed minimum progress* priority rule. At each node in each time slot, let *group* i be the set of packets at destination distance i. There are at most n such groups. In order from the smallest numbered group to the largest, assign exactly 1 packet from each nonempty group to an outgoing link along a shortest path to its destination. Following this, the assignment of the remaining packets to links is arbitrary. Indeed, all such packets may be purposefully deflected!

Proposition 4.1: Let a batch of N^0 packets be admitted to a binary n-cube. If packets are routed under the *guaranteed minimum progress* priority rule, the evacuation time satisfies

$$T \leq n + 2(N^0 - 1). \tag{4.1}$$

Proof: In the binary n-cube, a packet at destination distance i has exactly i outgoing links from the current node along a shortest path to its destination. Under the ordering of link assignments in the *guaranteed minimum progress* rule, the packet chosen from each group to be assigned to an outgoing link will have at least 1 unassigned, shortest path link available. Hence, one packet from each group, or equivalently from each destination distance, moves 1 step closer to its destination. Hence condition (2.2) is met, and applying *Lemma 2.1* yields (4.1). □

4.2. Batch Delay on Arbitrary Topology

We next derive an evacuation time bound for a batch of N^0 packets, each destined for one of M nodes in an arbitrary topology with diameter L and deflection index D_I. We begin by ordering the M destinations in an arbitrary fashion, and assume that this ordering is known to each node. Let N_j^0, $j = 1, 2, \ldots, M$, be the number of packets initially admitted and destined for the j^{th} destination, so $N^0 = \sum_{j=1}^{M} N_j^0$. We refer to packets destined for node j as *class j* packets. Consider the following priority packet-to-link assignment rule, which we call the *ordered destination* rule. Let each node order arriving packets by class from the lowest to the highest numbered class. Starting from the lowest numbered class, packets are assigned to a preferred output link, if available. The choice of ordering for packets within a class is arbitrary.

Let T_j be the evacuation time of class j packets, with $T_0 \equiv 0$. Let τ_j denote the time by which *all* classes up to and including the j^{th} class have evacuated, i.e. $\tau_j = \min \{ t : N_i^t = 0 \,,\, i = 1,2,...,j \}$. It follows that $\tau_1 \le \tau_2 \le \cdots \le \tau_M$ and $T_j \le \tau_j$. Let *phase j* refer to the interval $[\tau_{j-1}, \tau_j)$. Note that during phase j, all packets of class $k < j$ have been evacuated. Further, under the *ordered destination* rule, packets of class j can not be deflected by packets of class $k > j$. Hence, we may view each phase as a separate evacuation period for packets sent to a single destination. We may apply the single destination evacuation time bound (3.1) to upper bound the length of each phase $j = 1,2,...,M$,

$$\tau_j \le \tau_{j-1} + L + D_I(N_j^{\tau_{j-1}} - 1) \le \tau_{j-1} + L + D_I(N_j^0 - 1). \tag{4.2}$$

It follows that the network evacuates in time

$$T = \tau_M = \sum_{j=1}^{M} \tau_j - \tau_{j-1} \tag{4.3}$$

$$\le \sum_{j=1}^{M} L + D_I(N_j^0 - 1) = LM + D_I(N^0 - M). \tag{4.4}$$

Remark: Eq. 4.4 is useful since it represents an evacuation time bound which is a function of the number of packet destinations, not just the topology parameters L and D_I. For networks with $D_I = L$ (e.g., "go-back-all" network), (4.6) is identical to (H2).

By choosing the ordering of destinations appropriately, the multiple destination evacuation time bound (4.4) can be considerably reduced. Indeed, the following proposition indicates that the evacuation time bound is minimized when a walk connecting destinations in order has minimum length.

Proposition 4.2: Let a batch of N^0 packets be admitted to a "go-back" network with deflection index D_I and diameter L. Suppose each packet is destined for one of M destination nodes, and packets are assigned to links by the *ordered destination* rule. Then, the N_j^0 packets of the j^{th} class evacuate in time

$$T_1 = \tau_1 \le L + D_I(N_1^0 - 1), \tag{4.5a}$$

$$T_j \le \tau_j \le \max \left[L, \{ \max_{k : 1 \le k \le j-1} (\tau_k + \rho_{k,j} + D_I) \} + D_I(N_j^0 - 1) \right] \qquad j = 2,...,M, \tag{4.5b}$$

where $\rho_{k,j} \triangleq$ *distance from* k^{th} *to* j^{th} *destination node*. If the destinations are ordered so a walk connecting destinations in order has length P, then the network evacuation time satisfies

$$T \le L + P + D_I\,(N^0 - 1). \tag{4.6}$$

Proof: The details of the proof are omitted, though we develop the result as follows. Apply Proposition 3.1 to the "batch" of packets comprising the first class, and (4.5a) follows immediately. Now consider the packets in all other classes with $j > 1$. Under the *ordered destination* rule, packets of class j can be deflected by packets of class $k \le j$. The first class j packet exits no later than $\rho_{k,j} + D_I$ hops after the evacuation of all packets up to and including the k^{th} class (i.e., at time τ_k), for the value of $k \le j-1$ for which $\tau_k + \rho_{k,j}$ is largest. At least one of each of the remaining class j packets exits in each subsequent interval of D_I time slots, until the j^{th} class is evacuated. Hence the j^{th} class must evacuate in time given by

(4.5b). Summing (4.5a) and (4.5b) for all $j = 2,...,M$ and applying the triangle inequality yields (4.6).□

Remarks: To minimize the bound on evacuation time, we choose the ordering of destinations to minimize the walk length P. Note that in many regular networks of $|V|$ nodes it is possible to span all nodes in a walk of length $P = |V| - 1$. For networks with $D_I \ll L$, (4.6) is a considerable improvement over (H2). For the n-cube with a single destination demand, (4.6) is identical to (H3). For multiple destination demands on a fully loaded n-cube with $P = |V| - 1$, (4.6) is approximately a factor of $\frac{n + 1/2}{n}$ larger than (H3).

4.3. Continuous Admissions to the Hypercube

In this section we study the maximum transit delay of packets admitted continuously to a binary n-cube. Once again we route packets according to the *guaranteed minimum progress* priority rule, with a small modification to avoid livelock under continuous admissions. As before, at each node in each time slot let *group i* be the set of packets at destination distance *i*. In order from the smallest numbered group to the largest, assign the **oldest** packet (i.e., longest in the network) from each nonempty group to an outgoing link along a shortest path to its destination. (If several packets in a group are equally "old" one is picked arbitrarily). As before, the assignment of all remaining packets to links is arbitrary.

Proposition 4.3: Suppose packets are admitted continuously to a binary n-cube, with packets routed under the *modified* guaranteed minimum progress priority rule (as described above). Let N^τ denote the number of packets in the network at the end of time slot τ. The maximum transit delay ($T(\tau)$) of a packet admitted during time slot τ is

$$T(\tau) \le n + 2(N^\tau - 1). \tag{4.7}$$

The maximum delay of a packet admitted in any time slot (T) is bounded by

$$T \le n + 2(n2^n - 1). \tag{4.8}$$

Proof: Consider the N^τ packets in the network at the end of time slot τ. Under our routing scheme, a packet from each nonempty group (i.e., at each destination distance) from among the N^τ moves closer to its destination distance in every time slot. We may view the initial collection of N^τ packets as a "batch" of packets effectively admitted at time τ. Since routing these packets under the guaranteed minimum progress rule assures that we meet condition (2.2), all packets in this "batch" must be routed within a time given by (4.1). Hence, the delay of the longest delayed packet admitted at time τ is bounded in (4.7). □

Discussion: Suppose all packets in a fully populated n-cube are destined for a single node. Then each of nearly $n\ 2^n$ packets must traverse one of the n links incident to the destination node. This indicates that a lower bound on worst case delay is $O(2^n)$. Note that the upper bound (4.8) is only $O(n)$ times larger than this lower bound.

5. Conclusion

We have derived several new bounds on the delay of packets admitted to networks operating with deflection routing. We have considered both single and multiple destination demands. For multiple destination demands, no assumptions were made which limit the number of packets to be routed, nor the distribution of destinations. Indeed, the bounds are valid even for demands which may represent a malicious cooperation between network nodes.

Most of our bounds can be shown to be tight for small numbers of admitted packets. For larger numbers of admitted packets the bounds might be quite loose. We note that a number of authors have published simulation results for evacuation time and maximum delay for deflection routing on various topologies under uniform offered loads [2, 7, 9, 10, 11]. These simulations all indicate that deflection routing yields remarkably low *mean* transit delay for heavily loaded networks with uniform offered traffic. Indeed, observed worst case transit delays are often only a fraction of the bounds presented in this paper. Less has been reported in the literature regarding worst case delays produced by highly nonuniform demands [8, 12, 13, 14] where we would expect to observe the longest transit delays.

There appear to be several opportunities for further research on worst case delay. One is to consider limits on the number of packets admitted for each destination. As an example, Borodin and Hopcroft conjecture that some adaptive "deflection" routing rule exists which can route a permutation on a binary n-cube in $O(n^k)$ time [15]. This suggests that a deflection routing rule may exist which can route a permutation faster than the $O(n^2)$ time required by sorting algorithms [16]. A second opportunity is to consider probabilistic bounds. Use of such bounds may be particularly helpful in estimating the required capacity of resequencing buffers.

References

1. N. F. Maxemchuk, "The Manhattan Street Network," *IEEE Globecom 1985*, pp. 255-261, New Orleans, La., September, 1985.
2. J. T. Brassil and R. L. Cruz, "An Approximate Analysis of Packet Transit Delay in the Manhattan Street Network," *submitted to* IEEE INFOCOM'92, June, 1991.
3. D. H. Lawrie and D. A. Padua, "Analysis of Message Switching with Shuffle-Exchanges in Multiprocessors," *Proceedings of Workshop on Interconnection Networks for Parallel and Distributed Processing*, pp. 116-123, ACM, April, 1980.
4. B. Hajek, "Bounds on Evacuation Time for Deflection Routing," *Distributed Computing*, vol. 5, no. 1, pp. 1-5, 1991. Also appeared in *Proceedings of CISS'89*, March, 1989.
5. S. A. Felperin, L. Gravano, G. D. Pifarre, and J. L. C. Sanz, "Routing Techniques for Massively Parallel Communication," *Proceedings of the IEEE*, vol. 79-1, no. 4, pp. 488-503, April, 1991.
6. N. F. Maxemchuk, "Regular Mesh Topologies in Local and Metropolitan Area Networks," *AT&T Technical Journal*, vol. 64, no. 7, pp. 1659-1685, September, 1985.
7. A. G. Greenberg and B. Hajek, "Deflection Routing in Hypercube Networks (3/89 Draft)," *submitted to* IEEE Trans. on Communications, 1989.
8. J. Y. Ngai and C. L. Sietz, "A Framework for Adaptive Routing in Multicomputer Networks," *1989 ACM Symposium on Parallel Algorithms and Architectures*, 1989.
9. A. Krishna and B. Hajek, "Performance of Shuffle-Like Switching Networks with Deflection," *Proceedings of IEEE INFOCOM'90*, pp. 473-480, June, 1990.
10. A. G. Greenberg and J. W. Goodman, "Sharp Approximate Models of Adaptive Routing in Mesh Networks (Preliminary report)," *Teletraffic Analysis and Computer Performance Evaluation*, Elsevier Science Publishers, B. V., 1986.
11. T. Syzmanski, "An Analysis of Hot-Potato Routing in a Fiber Optic Packet Switched Hypercube," *IEEE INFOCOM '90 Proceedings*, vol. 1, pp. 918-925, June, 1990.
12. J. T. Brassil and R. L. Cruz, "Nonuniform Traffic in the Manhattan Street Network," *Proceedings of ICC'91*, vol. 3, pp. 1647-1651, June, 1991.
13. N. F. Maxemchuk, "Problems Arising from Deflection Routing: Live-lock, Lockout, Congestion and Message Reassembly," *Proceedings of NATO Workshop on Architecture and High Performance Issues of High Capacity Local and Metropolitan Area Networks*, France, June, 1990.
14. A. K. Choudhury and N. F. Maxemchuk, "Effect of a Finite Reassembly Buffer on the Performance of Deflection Routing," *Proceedings of ICC'91*, vol. 3, pp. 1637-1646, June, 1991.
15. A. Borodin and J. E. Hopcroft, "Routing, Merging and Sorting on Parallel Models of Computation," *Journal of Computer and System Sciences*, vol. 30, pp. 133-145, 1985.
16. K. E. Batcher, "Sorting Networks and their applications," *AFIPS Spring Joint Computer Conference*, vol. 32, pp. 307-314, 1968.

Large Deviations of Traffic in High Speed Digital Networks with a View Towards Network Control [1]

G. Kesidis and J. Walrand
EECS Dept., Cory Hall, University of California, Berkeley, CA94720.

Abstract

We use the theory of large deviations to analyze loss probability in a given buffer due to buffer overflow. The buffer is assumed to have a deterministic server (fiber or switch fabric) and bursty input sources. Markov fluid and batch Poisson models of bursty traffic are considered. The expressions for loss probability derived here can be applied to the problem of call acceptance and routing in high speed digital networks [1].

1 Introduction

In future high speed digital networks, the traffic due to one call may be bursty. Several models have been proposed to describe bursty traffic: batch Poisson, Markov fluid, and Markov modulated Poisson processes. In this paper we use the theory of large deviations to analyze cell loss in network buffers with bursty sources.

The call set-up and routing algorithm described in [1] requires estimates of the capacity of each buffer in the network to accept more calls. Let $F(N, B, c)$ be the fraction of cells lost due to buffer overflow of a given buffer of size B cells, with deterministic service rate c cells/s, and N sources (calls). Using the theory of large deviations, it was shown in [1] that, for small $\epsilon > 0$, $F(N(1+\epsilon), B, c) \approx F(N, B, \frac{c}{1+\epsilon})$; to each buffer, a "virtual" buffer process was added with the same N inputs as the original buffer but with service rate $\frac{c}{1+\epsilon}$. Actually, in order to reduce the variance of the estimate of $F(N, B, \frac{c}{1+\epsilon})$, more than one virtual buffer process was proposed – each with a smaller buffer size (importance sampling).

In section 2 an expression for the probability of buffer overflow in a busy cycle, $\Phi(N, B, c)$, is derived assuming the sources are statistically identical. Two models of bursty traffic are considered: Markov fluid and batch Poisson. In section 3 we use Φ to derive an expression for $F(N, B, c)$ for the Markov fluid case. In section 4 we assume the traffic characteristics are known and describe a way of quickly estimating Φ by simulation (based on ideas in [2] and [3]). Conclusions are drawn in section 5.

2 Probability of Buffer Overflow in a Busy Cycle

2.1 Markov Fluid Sources

Markov fluids are commonly used to model bursty ATM traffic [4]. In this paper we consider an m-rate Markov fluid model: each call generates cells at one of m different deterministic rates. The time derivative of each call is an m state birth-death process. We consider a given buffer of size B cells, with N m-rate Markov fluid calls sharing the

---

[1] Work supported by: NSERC of Canada, Pacific Bell, Micro Grant of the State of California, ARO Grant Number DAAL03-89-K-0128

buffer, and served by a fiber with transmission rate c cells/s. In this section we consider calls of the same type to derive bounds on the probability of buffer overflow in a busy cycle.

Define the buffer occupancy Y by

$$\dot{Y}(t) \; = \; \sum_{k=1}^{N} R_k(t) - c \; = \; \sum_{k=1}^{N} (R_k(t) - \frac{c}{N})$$

where R_k is the right-hand time derivative of the k^{th} input to the buffer. The R_k's are i.i.d. m-state birth-death Markov chains with: state space $\underline{\Lambda} = \{\Lambda_1, \Lambda_2, ..., \Lambda_m\}$, $\Lambda_i > \Lambda_{i-1}$, transition rate matrix Q, and stationary distribution $\underline{\pi}$. Therefore Y has piecewise linear trajectories with Markov slope. Letting T_Y^b be the first time that the process Y hits level b, we write:

$$\Phi(N, B, c) \; = \; \mathrm{P}(T_Y^{B+1} < T_Y^0).$$

To compute this probability, the restriction $Y \geq 0$ is not necessary. We thus refer to Y as the *free process*.

We make the following assumptions and definitions in order to apply the large deviations result in [5] to get bounds on Φ:

[A1] $N\Lambda_m > c$ (this assumption is necessary else $Y(t) \leq 0$ eventually).

[A2] The traffic intensity $\rho = N < \underline{\pi}, \underline{\Lambda} > /c < 1$ (the stability condition). Here $< \underline{x}, \underline{z} >$ denotes inner product between vectors $\underline{x}$ and $\underline{z}$.

[A3] The drift of $\dot{Y}(t)$ is negative whenever $\dot{Y}(t) \geq 0$. This may be somewhat restrictive in the range of allowable model parameters.

Define the cumulant ([6], p.145) of the process R_k as the function $H(r, \alpha)$ such that $\exp\{\alpha R_k(t) - \int_0^t H(R_k(s), \alpha) ds\}$ is a martingale. We find that

$$H(r, \alpha) \; = \; \lambda(r)(e^{\alpha \Delta^{+}(r)} - 1) + \mu(r)(e^{-\alpha \Delta^{-}(r)} - 1)$$

where λ is piecewise linear on $[\Lambda_1, \Lambda_m]$ (PWL) such that $\lambda(\Lambda_m) = 0$ and $\lambda(\Lambda_i) = Q[\Lambda_i, \Lambda_{i+1}]$ for $1 \leq i \leq m-1$. Similarly, μ is PWL with $\mu(\Lambda_1) = 0$ and $\mu(\Lambda_i) = Q[\Lambda_i, \Lambda_{i-1}]$ for $2 \leq i \leq m$. Δ^{+} is PWL with $\Delta^{+}(\Lambda_m) = 0$ and $\Delta^{+}(\Lambda_i) = \Lambda_{i+1} - \Lambda_i$ for $1 \leq i \leq m-1$. Similarly, Δ^{-} is PWL with $\Delta^{-}(\Lambda_1) = 0$ and $\Delta^{-}(\Lambda_i) = \Lambda_i - \Lambda_{i-1}$ for $2 \leq i \leq m$. Note that $H(r, 0) = 0$ for all r, where r is now allowed to range continuously between the smallest source rate Λ_1 and the maximum source rate Λ_m.

Define the Legendre transform of H:

$$H^*(r, s) \; = \; \sup_{\alpha \in \mathbf{R}} \{s\alpha - H(r, \alpha)\}.$$

H^* is non-negative, strictly convex in (r, s), and $H^*(r, s) = 0$ if and only if $s = \frac{\partial H}{\partial \alpha}(r, 0) =$ the drift of R_k when $R_k = r$.

Note that $\dot{Y}$ is a pointwise function of N Markovian processes (but Y itself is not).

Thus we can apply the large deviations results in [6] (chap. 5) to $\dot{Y}$. Let

$$Z^B(t) \;=\; \frac{\dot{Y}((B+1)t)}{B+1}$$

and observe that

$$\Phi(N,B,c) = \mathrm{P}\left(\int_0^t Z^B(s)ds \text{ hits } \frac{1}{B+1} \text{ before it hits } 0\right). \qquad (1)$$

With the contraction mapping principle in mind [7], we define (z corresponds to $\dot{Y}$ and ζ corresponds to $\ddot{Y}$)

$$\begin{aligned} L^*_{N,c}(z,\zeta) \;&=\; \inf_{<\underline{y},\underline{1}>=z,<\underline{s},\underline{1}>=\zeta} \sum_{j=1}^{N} H^*(y_j + \frac{c}{N}, s_j) \\ &=\; NH^*\left(\frac{z+c}{N}, \frac{\zeta}{N}\right) \text{ , by convexity and symmetry,} \end{aligned}$$

where $\underline{r} = (r_1, ..., r_N)$ and $\underline{1} = (1, ..., 1)$. Thus, $L^*_{N,c}$ is non-negative and convex too. Note that the drift of $\dot{Y}$ when $R_1 = r_1, \ldots, R_N = r_N$ is $\sum_{k=1}^{N} \frac{\partial H}{\partial \alpha}(r_k, 0)$. So by assumption **A3**

$$\sum_{k=1}^{N} \frac{\partial H}{\partial \alpha}(r_k, 0) \;<\; 0$$

whenever $< \underline{r}, \underline{1} > \geq c$. In particular, for all $r \geq c/N$, **A3** implies $\frac{\partial H}{\partial \alpha}(r, 0) < 0$.

With (1) in mind, define the following set of trajectories of Z^B:

$$\Gamma^*_B(T) \;=\; \{z \in C^1[0,T] \;:\; \int_0^T z(t)dt = (B+1)^{-1} > \int_0^u z(t)dt > 0 \;\; \forall u \in (0,T)\}.$$

Note that this set depends on the parameter B. By the large deviations result in [5] we get the *upper bound*,

$$\Phi(N,B,c) \leq \exp(-(B+1)I_B(N,c) + o(B)) \qquad (2)$$

where

$$I_B(N,c) \;=\; \inf_{T > \frac{1}{N\Lambda_m - c}} \; \inf_{z \in \Gamma^*_B(T)} \int_0^T L^*_{N,c}(z(t), \dot{z}(t))dt.$$

Note that $\frac{B+1}{N\Lambda_m - c}$ is the shortest possible time to overflow. Since $L^*_{N,c} \geq 0$, we can replace $\Gamma^*_B(T)$ in the expression for $I_B(N,c)$ by

$$\Gamma_B(T) \;=\; \{z \in C^1[0,T] \;:\; \int_0^T z(t)dt = (B+1)^{-1}, z(0) = 0, z(t) > 0 \;\; \forall t \in (0,T)\}.$$

By Jensen's inequality, for all $z \in \Gamma_B(T)$

$$\int_0^T L^*_{N,c}(z(t), \dot{z}(t))dt \;\geq\; TL^*_{N,c}\left(\frac{1}{T}\int_0^T z(t)dt, \frac{1}{T}\int_0^T \dot{z}(t)dt\right) \;=\; TL^*_{N,c}\left(\frac{1}{(B+1)T}, \frac{z(T)}{T}\right)$$

$$= TNH^*\left(\frac{1}{(B+1)TN}+\frac{c}{N},\frac{z(T)}{TN}\right).$$

Claim: $H^*(r,s) \geq H^*(r,0)$ when $r \geq \frac{c}{N}$ and $s > 0$.
Proof: For all $r \geq \frac{c}{N}$ and $s > 0$

$$\begin{aligned} H^*(r,s) &\geq \sup_{\alpha>0}\{s\alpha - H(r,\alpha)\} \\ &\geq \sup_{\alpha>0}\{-H(r,\alpha)\} \quad \text{since } s>0 \\ &= H^*(r,0) \quad \text{by the lemma below.} \end{aligned}$$

Lemma: $\arg\sup_{\alpha\in\mathbf{R}}\{-H(r,\alpha)\} > 0$ for all $r \geq \frac{c}{N}$.
Proof: Fix $r \geq \frac{c}{N}$ and note that $H(r,0)=0$, $\frac{\partial^2 H}{\partial\alpha^2}(r,0) > 0$, and $\lim_{|\alpha|\to\infty} H(r,\alpha) = \infty$. Recall that assumption **A3** implies $\frac{\partial H}{\partial\alpha}(r,0) < 0$. By the continuity of $H(r,\cdot)$ and the facts above we conclude that there exists a unique $\alpha_r > 0$ that minimizes $H(r,\cdot)$ (draw the graph of $H(r,\alpha)$ vs α), and the lemma is proved.

So by Jensen's inequality and the claim above,

$$\begin{aligned} \Phi(N,B,c) &\leq \exp\{-(B+1)I(N,c)+o(B)\} \\ &\leq \exp\left\{-\inf_{T>\frac{1}{N\Lambda_m - c}} (B+1)TNH^*(\frac{1}{(B+1)TN}+\frac{c}{N},0)+o(B)\right\} \\ &= \exp\left\{-\inf_{S>\frac{B+1}{\Lambda_m - c/N}} SH^*(\frac{1}{S}+\frac{c}{N},0)+o(B)\right\} \end{aligned}$$

where we have substituted $S=(B+1)NT$. Note that the straight line path, $z_s(t) \equiv \frac{1}{(B+1)T}$, is *not* in $\Gamma^*(T)$.

For $m=2$, assumption **A2** implies assumption **A3** and we can show explicitly that $H^*(\frac{c}{N}+\epsilon,0)$ is increasing in ϵ. Thus the minimizing S equals $(B+1)(\Lambda_m - \frac{c}{N})^{-1}$ so that

$$\begin{aligned} \Phi(N,B,c) &\leq \exp\left(-(B+1)\frac{H^*(\frac{\Lambda_m}{B+1}+\frac{c}{N}-\frac{c}{N(B+1)},0)}{\Lambda_m - \frac{c}{N}}+o(B)\right) \\ &= \exp\left(-B\frac{H^*(\frac{c}{N},0)}{\Lambda_m-\frac{c}{N}}+o(B)\right). \end{aligned}$$

In this special case the formula for $H^*(r,0)$, $r\in(\Lambda_1,\Lambda_2)$, is easy to obtain:

$$H^*(r,0) = \left(\sqrt{\mu(r)}-\sqrt{\lambda(r)}\right)^2.$$

For a lower bound on Φ we assume

[A4] as $B\to\infty$, $\frac{N}{B+1}\to\gamma$, where γ is a positive constant.

Consider the following trajectory of $\dot{Y}$:

$$z(t) = (N\Lambda_m - c)(1-e^{-t}).$$

The corresponding trajectory of $\dot{Z}^B$ is

$$z^B(t) \;=\; \frac{(N\Lambda_m - c)(1 - e^{-(B+1)t})}{B+1}.$$

By assumption **A4**

$$\frac{z^B(t)+c}{N} = \frac{\gamma\Lambda_m + c}{N} + o(\frac{1}{B}) \quad \text{and} \quad \frac{\dot{z}^B(t)}{N} = o(\frac{1}{B})\,.$$

Define T^B by $\int_0^{T^B} z^B(t)dt = \frac{1}{B+1}$ and note that $T^B \sim (N\Lambda_m - c)^{-1}$ as $B \to \infty$.
Using Wentsell's lower bound result ([6], chap. 5),

$$\begin{aligned} \Phi(N,B,c) &\gtrsim \mathrm{P}\{\dot{Y}^B \in \mathcal{B}_{T^B}(z^B, \delta_B)\} \\ &\geq \exp\left(-B(I_{T^B}(z^B) + \delta_B) + o(B)\right), \end{aligned}$$

where $\mathcal{B}_T(r,\delta)$ is a tube of radius δ, in the sup norm, about trajectory r on $[0,T]$, $\delta_B = o(\frac{1}{B})$, and

$$I_T(z) \;=\; \int_0^T NH^*\left(\frac{z(t)+c}{N}, \frac{\dot{z}(t)}{N}\right) dt.$$

Since $T^B \sim (N\Lambda_m - c)^{-1}$, we can use assumption **A4** above to get

$$\begin{aligned} BI_{T^B}(z^B) &= B\frac{NH^*(\frac{\gamma\Lambda_m + c}{N} + o(\frac{1}{B}), o(\frac{1}{B}))}{N\Lambda_m - c} + o(B) \\ &= B\frac{H^*(\frac{\gamma\Lambda_m + c}{N}, 0)}{\Lambda_m - \frac{c}{N}} + o(B). \end{aligned}$$

Thus

$$\Phi(N,B,c) \;\gtrsim\; \exp\left(-B\frac{H^*(\frac{\gamma\Lambda_m + c}{N}, 0)}{\Lambda_m - \frac{c}{N}} + o(B)\right).$$

2.2 Batch Poisson Sources

Alternatively, we can define the buffer process $Z(t) = A^N(t) - ct$ where $A^N = \sum_{i=1}^N A_i$ and the A_i are i.i.d. batch Poisson sources. That is,

$$A_i(t) \;=\; \sum_{j=1}^{N_i(t)} G_{i,j}$$

where the N_i are identically distributed Poisson process with rate ν and the $G_{i,j}$ are identically distributed random variables, taking values in $\{1, 2, ..., K\}$, and having mean $\bar{G}$. The random objects $\{N_i, G_{i,j};\ i = 1, \ldots, N;\ j = 1, 2, \ldots\}$ are mutually independent. We shall assume that the system is stable, i.e., that $N\nu\bar{G} < c$. Let T^b be the first hitting time of b by Z. Using the large deviations result in [7], we can find an expression for

$P\{T^{B+1} < T^0 | Z(0) = k\}$ and thereby get an expression for

$$\Phi(N, B, c) = \sum_{k=1}^{K} P\{T^{B+1} < T^0 | Z(0) = k\} P\{G_{i,j} = k\}. \tag{3}$$

The cumulant of A^N is given by

$$H_N(\alpha) \;=\; \log E e^{\alpha(A^N(1) - A^N(0))}.$$

That is. $\exp\{\alpha A^N(t) - H^N(\alpha)t\}$ is a martingale. Note that H_N is the log moment generating function of $A^N(1) - A^N(0)$ (because A_N has independent and stationary increments) which recalls Cramér's theorem for i.i.d. sums of random variables. Let $H_N^*(x) = \sup_{\alpha \in \mathbf{R}}\{x\alpha - H_N(\alpha)\}$ - the Legendre transform of H_N. Define the scaled process $A^{B-k+1,N}(t) = A^N((B-k+1)t)/(B-k+1)$. Using the large deviations result in [7] we get

$$P\{T^{B+1} < T^0 | Z(0) = k\} \;=\; \exp\left[-(B-k+1) \inf_{T>0} \inf_{a \in \Omega^*(T)} \int_0^T H_N^*(\dot{a}(t))dt + o(B)\right];$$

where the set of trajectories of $A^{B-k+1,N}$ that result in an overflow at time $(B-k+1)T$ in the first busy cycle of Z is
$\Omega^*(T) = \{a \in C^1[0,T] \mid a(0) = \frac{k}{(B-k+1)T},\ a(T) - a(0) = 1 + cT > a(t) - a(0) > -\frac{2k}{(B-k+1)} + ct\ \forall t \in (0,T)\}$.

Since H_N^* is convex we can use Jensen's inequality, together with the observation that H_N^* depends only on $\dot{a}$ and not on a, to get that the infimum over the set $\Omega^*(T)$ is achieved by a straight line path. Thus

$$\begin{aligned} P\{T^{B+1} < T^0 | Z(0) = k\} \;&=\; \exp\left[-(B-k+1) \inf_{T>0} \inf_{a \in \Omega^*(T)} T H_N^*\left(\frac{a(T) - a(0)}{T}\right) + o(B)\right] \\ &=\; \exp\left[-B \inf_{T>0} T H_N^*(\frac{1}{T} + c) + o(B)\right]. \end{aligned}$$

We should mention at this point that, owing to the fact that the large deviations theory is directly applicable to the *Markovian* process Z, the expression above is an *exact* asymptotic result. This is in contrast with equation (2) for the Markov fluid model, which was only an upper bound. Using the expression (3) for Φ, we get

$$\Phi(N, B, c) \;=\; \exp(-B \inf_{T>0} T H_N^*(\frac{1}{T} + c) + o(B)).$$

Note that H_N^* is convex, $H_N^* \geq 0$, and $H_N^*(x) = 0$ only if $x = NE[A_1(1) - A_1(0)] < c$ (recall the stability condition $N\nu\bar{G} < c$). Thus $H_N^*(x) > 0$ for all $x \geq c$ which implies that $\arg\inf_{T>0} T H_N^*(\frac{1}{T} + c) < \infty$.

Since the batch Poisson processes A_i are i.i.d., $H_N(\alpha) = N H_1(\alpha)$ which implies that $H_N^*(x) = N H_1^*(\frac{x}{N})$. Substituting this into the expression for Φ and changing variables $(S = NT)$ we get

$$\Phi(N, B, c) \;=\; \exp(-B \inf_{T>0} N T H_1^*(\frac{1}{NT} + \frac{c}{N}) + o(B))$$

$$
\begin{aligned}
&= \exp(-B \inf_{S>0} S H_1^*(\frac{1}{S} + \frac{c}{N}) + o(B)) \\
&= \Phi(1, B, \frac{c}{N}).
\end{aligned}
$$

3 Fraction of Cells Lost for Markov Fluid Sources

The fraction of cells lost due to buffer overflow, $F(N, B, c)$, can be expressed in terms of the average number of cells lost in an overflowing cycle, D, the average number of cells arriving in a cycle, C, and Φ:

$$F = \frac{\Phi D}{C}, \tag{4}$$

assuming that the rate processes are stationary. Note that the numerator, ΦD, is the average number of cells lost in a busy cycle.

In finding an expression for C, we ignore the buffer size B because the number of cells arriving in an overflowing cycle is of order B and the probability of an overflowing cycle is of order $\exp(-BI + o(B))$. This gives us a contribution to C of approximately $\exp(-BI + o(B))$ which is negligible for large B. In finding an expression for D we assume that the typical value of $\dot{Y}$ when an overflow occurs is $\gamma\Lambda_m = \frac{N\Lambda_m}{B+1}$ (see section 2.1). Also, we assume a negligible number of cells will be lost after $\dot{Y}$ returns to zero. Under these assumptions, D will have a negligible dependence on B as well (for large B). Thus $C/D = \exp(o(B))$.

4 Quick Simulation of Markov Fluids

In this section we assume that an estimate of the birth-death transition rate matrix (TRM) and the state space of the time derivative of each buffer source is available. We denote this TRM by Q^*. We want to simulate the buffer occupancy using another TRM, $\hat{Q}$ in order to obtain an accurate estimate of Φ quickly. Using $\hat{Q}$, an estimate of Φ under Q^* is obtained by computing likelihood ratios, L_i, for each cycle i [3]:

$$\Phi = \sum_{i=1}^{n} V_i L_i$$

where $V_i = 1\{\text{cells are lost in cycle } i\}$. We want a choice of $\hat{Q}$ that results in a much smaller variance of the estimate of Φ than that of a direct sample average estimate using Q^*.

For Markov fluid sources we have argued above that the typical value of the time derivative of the buffer occupancy process, $\dot{Y}$, at the time of overflow is $\gamma\Lambda_m$. This motivated us to choose $\hat{Q}$ so that the mean time derivative of each source is $\frac{1}{N}(\gamma\Lambda_m + c)$. In particular, we chose

$$\hat{Q} = \inf_{Q \in \Omega} J(Q|Q^*)$$

where $\Omega = \{Q \text{ a birth-death TRM} : \pi Q = 0,\ < \pi, \Lambda >= \frac{1}{N}(\gamma\Lambda_m + c),\ \pi << \pi^*\}$,

$\pi^* Q^* = 0$, and J is the relative entropy ([8] p.127,128),

$$J(Q|Q^*) \;=\; \sum_{i=1}^{m-1} \pi(\Lambda_i)\lambda(\Lambda_i)\log\left(\frac{\lambda(\Lambda_i)}{\lambda^*(\Lambda_i)}\right) + \sum_{i=2}^{m} \pi(\Lambda_i)\mu(\Lambda_i)\log\left(\frac{\mu(\Lambda_i)}{\mu^*(\Lambda_i)}\right).$$

$\hat{Q}$ will now be explicitly calculated in the two-rate case and a simulation result will be described. Let $\lambda = \lambda(\Lambda_1) = Q(\Lambda_1, \Lambda_2)$ and $\mu = \mu(\Lambda_2) = Q(\Lambda_2, \Lambda_1)$. Similarly, let $\lambda^* = \lambda^*(\Lambda_1) = Q^*(\Lambda_1, \Lambda_2)$ and $\mu^* = \mu^*(\Lambda_2) = Q^*(\Lambda_2, \Lambda_1)$. Henceforth we write J as a function of (λ, μ) because Q^* and the Λ_i are assumed known.

With a Lagrange multiplier w, we define:

$$K(\lambda, \mu, w) \;\equiv\; J(\lambda, \mu) + w\left(\frac{\mu}{\mu+\lambda}\Lambda_1 + \frac{\lambda}{\mu+\lambda}\Lambda_2 - \frac{\gamma\Lambda_2 + c}{N}\right).$$

Solving $\partial K/\partial\lambda = \partial K/\partial\mu = 0$ we get

$$2\mu = w\Delta + \sqrt{(w\Delta)^2 + 4\lambda^*\mu^*} \quad \text{and} \quad 2\lambda = w\Delta + \sqrt{(w\Delta)^2 + 4\lambda^*\mu^*},$$

where $\Delta = \Lambda_1 - \Lambda_0$. Substituting μ and λ into the constraint ($\partial K/\partial w = 0$) we get

$$w\Delta \;=\; (2y - \Lambda_1 - \Lambda_2)\sqrt{\frac{\lambda^*\mu^*}{(\Lambda_2 - y)(y - \Lambda_1)}}.$$

where $y \equiv \frac{1}{N}(\gamma\Lambda_2 + c) = \frac{\Lambda_2}{B+1} + \frac{c}{N}$. This solution makes sense if we assume **[A5]** $\frac{c}{N} > \frac{\Lambda_1+\Lambda_2}{2}$ $\Rightarrow 2y - \Lambda_1 - \Lambda_2 > 0$, and **[A6]** $\Lambda_2 > y$. Note that assumption **A2** implies **A6** for large enough B. We now have expressions for the entries in $\hat{Q}$.

To demonstrate the speed-up obtainable by using the optimal μ and λ above, instead of μ^* and λ^*, we chose the following parameters: $\Lambda_1 = 0$, $\Lambda_2 = 2000$ cells/s, $B = 500$ cells, $c = 15000$ cells/s, $\mu^* = 20\ \text{s}^{-1}$, $\lambda^* = 10\ \text{s}^{-1}$ and $N = 14$. These parameters were chosen so that bit rates would be in the Mbps range and overflow would be rare. Using the expressions above, we computed $\lambda = 15.3\ \text{s}^{-1}$ and $\mu = 13.1\ \text{s}^{-1}$ (the entries of $\hat{Q}$). The goal of our simulations was to compute Φ with a sample standard deviation of 10%. The cost of a simulation was taken to be the number of transitions of the embedded Markov chain (an $N+1$ state birth-death process) required to reach the goal.

Using direct sample averaging (Q^*) we reached the goal in about 16×10^6 transitions with a value of $\Phi = 2.2 \times 10^{-4}$. Using $\hat{Q}$ and likelihood ratios we reached the goal in about 4×10^4 transitions with a value of $\Phi = 1.8 \times 10^{-4}$. This is a speed-up of approximately 400 times. "Trivial" busy cycles in which a cell arrives to an empty buffer and departs before the next cell arrives ($-c < \dot{Y} \leq 0$) where not counted.

In order to check that $(\lambda, \mu) = (15.3, 13.1)\text{s}^{-1}$ is optimal, we measured the sample standard deviation in 3×10^4 transitions using $(\lambda, \mu) \in \{(15.3 + \epsilon, 13.1 + \delta)\text{s}^{-1} : \epsilon, \delta \in \{-1, 0, 1\}\}$. An average was taken over 10 trials for each instance of (λ, μ). The results are shown in the following table: $(\lambda, \mu) = (15.3, 13.1)\text{s}^1$ (ie. $\epsilon = 0 = \delta$) resulted in the smallest sample standard deviation.

std.dev.[%]	δ=-1	δ=0	δ=1
ϵ=-1	15.1	15.0	16.0
ϵ=0	15.6	14.8	16.1
ϵ=1	16.7	15.4	16.6

5 Discussion and Conclusions

If the above approach to quick simulation is used for Markov fluid sources with more than two rates ($m > 2$), then the expression $\Phi = \sum V_i L_i$ no longer holds because the busy cycle start times are no longer regenerative. That is, the trajectories of the buffer occupancy over consecutive cycles are not independent. The probability of buffer overflow for the altered process, Φ^0, and the likelihood ratio of the evolution of the buffer occupancy over the *entire* simulation, L, must be computed because $\Phi = \Phi^0 L$. The variance of the estimate of Φ is now more difficult to compute[9].

In summary, the theory of large deviations was used to obtain bounds on the probability of buffer overflow in a busy cycle, Φ, with a deterministic server. We considered both Markov fluid and batch Poisson sources. Given Φ, we heuristically derived an expression for the fraction of cells lost due to buffer overflow for Markov fluid sources. Finally, we used these results to perform a quick simulation to evaluate Φ for Markov fluid sources.

Acknowledgements

We thank Dr. P. Konstantopoulos and G. de Veciana for their help in preparing this paper.

References

[1] C. Courcoubetis, G. Kesidis, A. Ridder, J. Walrand, and R. Weber, "Admission control and routing in ATM networks using inferences from measured buffer occupancy," *submitted for publication.*

[2] M. Cottrell, J.-C. Fort, and G. Malgouyres, "Large deviations and rare events in the study of stochastic algorithms," *IEEE Trans. on Auto. Control*, vol. 28 No. 9, pp. 907–920, 1983.

[3] S. Parekh and J. Walrand, "A quick simulation method for excessive backlogs in networks of queues," *IEEE Trans. on Auto. Control*, vol. 34 No. 1, pp. 54–66, 1989.

[4] M.Decina, T.Toniatti, P.Vaccari, and L.Verri, "Bandwidth assignment and virtual call blocking in atm networks," *IEEE Infocom'90*, vol. Vol.3, pp. 881–888, 1990.

[5] R. Dupuis, R. Ellis, and A. Weiss, "Large deviations for Markov processes with discontinuous statistics 1: General upper bounds," *preprint.*

[6] M. Freidlin and A. Wentsell, *Random Perturbations of Dynamical Systems.* New York, NY: Springer-Verlag, 1984.

[7] J. Lynch and J. Sethuraman, "Large deviations for processes with independent increments," *Annals of Probability*, vol. 15 No. 2, pp. 610–627, 1987.

[8] J.-D. Deuschel and D. W. Stroock, *Large Deviations.* New York, NY: Academic Press, 1989.

[9] S. M. Ross, *A Course in Simulation.* New York, NY: Macmillan, 1990.

A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks

Abhay K. Parekh and Robert G. Gallager
Laboratory for Information and Decision Systems
MIT, Cambridge MA 02139

We investigate the use of a generalized processor sharing multiplexing discipline combined with leaky bucket admission control, to provide performance guarantees to the users of an integrated services network. In such networks, many of the services (such as voice and video) are likely to have stringent delay requirements on *each packet* submitted. Thus, just providing average performance guarantees may not be adequate for these services. We assume a virtual circuit, connection-based packet network.

The multiplexing function is difficult in integrated services networks, since it has to satisfy the highly diverse requirements of many contending users. Performance guarantees imply that every session is assured a certain portion of the links it traverses. However, explicit reservations (through time or frequency division multiplexing) may be too wasteful to be practical. In addition to being efficient and flexible, the multiplexing discipline must be analyzable, so that performance guarantees can be made. We will present such a multiplexing discipline, and relate it to performance guarantees when the sources are regulated by leaky buckets.

A Generalized Processor Sharing (GPS) server is characterized by N positive real numbers, $\phi_1, \phi_2, ..., \phi_N$. The server works at a fixed rate r and is work conserving (i.e. it is never idle if there is work in the system). Let $S_i(\tau, t)$ denote the amount of session i traffic served in an interval $[\tau, t]$. Then a GPS server is defined as one for which

$$\frac{S_i(\tau,t)}{S_j(\tau,t)} = \frac{\phi_i}{\phi_j} \tag{1}$$

for any sessions, i, j and any interval $[\tau, t]$ for which both sessions have positive queue lengths. GPS is an attractive multiplexing scheme for a number of reasons:

- Define ρ_i to be the service rate guaranteed to session i. Then as long as $\rho_i < \frac{\phi_i}{\sum_j \phi_j} r$, the service rate guarantee to session i is met, independent of the demands of the other sessions.
- The delay of a session i packet can be bounded as a function of the session i queue length, independent of the queues and arrivals of the other sessions. Schemes such as FCFS, LCFS, and Strict Priority do not have this property.
- By varying the ϕ_i's we have the flexibility of treating the sessions in a variety of different ways. For example, when all the ϕ_i's are equal the system reduces to uniform processor sharing. As long as the combined average rate of the sessions is less than r, any assignment of positive ϕ_i's yields a stable system.
- It is possible to make worst case network queueing delay *guarantees* when the sources are constrained by leaky buckets. We will present our results on this later. Thus GPS is particularly attractive for sessions sending real-time traffic such as voice and video.

We now present a simple packet-by-packet transmission scheme that is an excellent approximation to GPS even when the packets are of variable length. Our idea is similar to the one used in [DKS89] to simulate bit-wise round robin. *We adopt the convention that a packet has arrived only after its last bit has arrived.* Let F_p be the time at which packet p will depart (finish service) under generalized processor sharing. We propose a scheme that closely approximates GPS as follows: Conceptually, the server simulates a GPS system on the packets that have already arrived—this simulation helps the server decide which packet to serve next, and proceeds independently from the actions actually taken by the server. At each time τ that a packet transmission is completed, the server chooses the first subsequent packet that would complete service in the GPS simulation. We call this scheme PGPS for *packet-by-packet* generalized processor sharing. Let $\hat{F}_p$ be the time at which packet p departs under PGPS. We have shown [PG91] that

Theorem 1 *For all packets p,*

$$\hat{F}_p - F_p < \frac{L_{max}}{r},$$

where L_{max} is the maximum packet length, and r is the rate of the server.

Having explained the multiplexing scheme, we turn to input regulation: Every source is constrained by the following leaky bucket scheme: The bucket size for session i is σ_i and the token arrival rate is ρ_i. The rate at which traffic enters the network is also constrained to be at most C_i bits/sec. If $A_i(\tau, t)$ is the amount of session i flow that leaves the leaky bucket and enters the network in the time interval $(\tau, t]$, then

$$A_i(\tau, t) \leq \min\{(t-\tau)C_i, \sigma_i + \rho_i(t-\tau)\}, \quad \forall t \geq \tau \geq 0, \tag{2}$$

for every session i. We say that session i conforms to (σ_i, ρ_i, C_i), or $A_i \sim (\sigma_i, \rho_i, C_i)$. It is assumed that the values of these parameters are set after negotiation with the user. This model for incoming traffic is very similar to the one recently proposed by Cruz [Cru91].

Consider a network of M GPS servers that is shared by leaky bucket regulated sessions. Session i is such that $A_i \sim (\sigma_i, \rho_i, C_i))$. Let $I(m)$ be the set of sessions that share server m and let the allocation of session $i \in I(m)$ be ϕ_i^m. We will assume that $\sum_{i \in I(m)} \rho_i < r^m$, where r^m is the rate of the server, m. For every session i define D_i^* to be the worst case delay (or time spent in the network) that could be faced by a session i packet over all possible feasible (leaky bucket constrained) sources. Similarly let Q_i^* be the largest number of session i bits in the network possible over the duration of session i's existence.

We have analyzed a broad class of GPS networks for which an important special case is the following: for each session i, and server m in the session i route, $\phi_i^m = \phi_i$. We have shown that every such GPS network is stable (whether acyclic or not), and can provide upper bounds on D_i^* and Q_i^* for every session i. Under this class of assignments, almost constant rate sessions with very stringent delay requirements can be assigned large values of ϕ, with minimal performance degradation to the other sessions.

Our analysis treats the the session i route as a whole. This yields bounds on D_i^* and Q_i^* that are significantly better than those obtained by treating each server in the route independently. Since the system is highly coupled, our bounds are not given in closed form (except for some simple, yet important cases), but a graphical algorithm is provided to compute them efficiently. Finally, if there are K_i links in the session i route, our GPS delay bound is never more than K_i packet transmission times greater than the worst case delay under PGPS.

It is hoped that these efforts will culminate in a scheme for resource assignment in high speed networks, that will help in establishing connections for sessions with stringent performance requirements.

References

[Cru91] Rene L. Cruz. A calculus for network delay, Part I: Network elements in isolation. *IEEE Transactions on Information Theory*, 37(1):114–131, January 1991.

[DKS89] Alan Demers, Srinivasan Keshav, and Scott Shenkar. Analysis and simulation of a fair queueing algorithm. *Proceedings of SIGCOMM '89*, pages 1–12, September 1989.

[PG91] A. K. Parekh and R. G. Gallager. A Generalized Processor Sharing approach to flow control—The single server case. Technical Report 2040, Laboratory for Information and Decision Systems, MIT, 1991.

Bandwidth and Buffer Allocation in ATM

S. Low and P. Varaiya *
University of California, Berkeley CA 94720

October 1, 1991

Abstract

We consider an ATM network modeled by output-buffered switches connected by links with fixed maximum transmission rates. The network provides R types of services. A unit of service of type r worths w_r, and is produced by a connection over a virtual circuit with fixed bandwidth μ_r and buffers b_r allocated to that connection along its path. We seek optimal bandwidth and buffer allocation to maximize revenue. We show that the problem is equivalent to a two-player nonzero sum game for which the optimal allocation always exists. The solution of the game suggests an implementation scheme which does not require knowledge of the traffic characteristics by the network.

1 Introduction

For our purposes an ATM network provides a one-way connection between two end points over a virtual circuit. A virtual circuit is a path between its end points and spanning several switching nodes. It is used to transport a message from one end point to the other. The message is packaged into small, uniformly sized units called cells. The bandwidth, i.e. number of cells per second or cps, allocated to a connection may vary from node to node and over the lifetime of the connection. If at any time the rate of input cells exceeds the bandwidth allocated to the connection, the node buffers the excess cells. The bandwidth at a node is shared among the active connections according to the service discipline at the node.

Recently, attention has been paid to the design of service disciplines that guarantee a fixed bandwidth to a connection regardless of changes in the number and burstiness of active connections [1, 2, 3]. [1] See [4] for a recent survey. This is motivated by the need to support real-time services [5, 6]. We call this class of service disciplines the fixed rate discipline, and a node that implements such discipline a fixed rate server. Experimental

---

*Work supported by Pacific Bell, the MICRO program.

[1] Although some of these works are concerned with store and forward packet networks, their discussions apply to ATM connections as well.

ATM networks of fixed rate servers, e.g. the Xunet [7, 1], are being built. The fixed rate discipline makes buffer management along the path straightforward, as we shall see below.

During connection setup, a virtual circuit reserves a fixed bandwidth at each node on its route. This bandwidth is then guaranteed to that virtual circuit for its lifetime and is not shared among other virtual circuits even when the message rate drops below the reserved bandwidth. Buffers are also reserved at each node to account for the fluctuation in the instantaneous message rate. If insufficient resources are available, the connection request is rejected. To determine the amount of bandwidth and storage that should be allocated at each node along the path of a connection is nontrivial. It generally depends on both the application requirements and resource availability in the network. In this paper we address the issue from the network provider's point of view and essentially ignore the application requirements. The network provides R types of services. A unit of service of type r worths w_r, and is produced by a connection over a virtual circuit with fixed bandwidth μ_r and buffers b_r allocated to that connection along its path. We seek optimal bandwidth and buffer allocation to maximize network revenue.

In §2 we propose a connection model. We model a message, or the cell stream over a connection, as a deterministic fluid. The service discipline at a node defines a mapping Φ from input messages to output messages. A connection spanning nodes $1, ..., k$ is modeled by the composite map $\Phi_k \circ \circ \circ \Phi_1$ where Φ_i is the map defined by node i. We show that the end to end behavior of a connection is equivalent to that of a single node - the bottleneck node. The result illustrates the tradeoff between bandwidth and buffers as a substitutable resources for providing services.

We investigate in §3 the bandwidth and buffer allocation problem that motivates this study. Based on the concept of burstiness curve [8], we propose in §3.1 a buffer reservation scheme in a network of fixed rate servers. §3.2 defines precisely optimal allocation in the network and show that the problem is equivalent to a two-player nonzero sum game. An optimal allocation exists, and is a saddle point when w_r is independent of r. The solution of the game suggests an implementation scheme when the traffic burstiness is known only to individual connections but not to the network. This is discussed in §3.3.

2 A Connnection Model

We call the cell stream over a connection a message. A message $m(t)$, $0 \leq t \leq T$, is a non-negative funtion which gives, at any time t, the instantaneous rate of the stream measured in cells per second or cps, and $T < \infty$ is its duration. In this section, we study the behavior of a virtual circuit spanning several fixed rate servers, to be defined below. It will be necessary to describe more precisely what a node does; see figure 1.

A node models a nonblocking output-buffered switch [9]. It consists of two parts: a switch and a node processor. The switch, for each virtual circuit, routes cells coming over the input links to the appropriate output links. In the node processor, there is a separate buffer and transmission resource for each output link. The transmission rate of each output link is fixed. Several virtual circuits may share the same output link, and

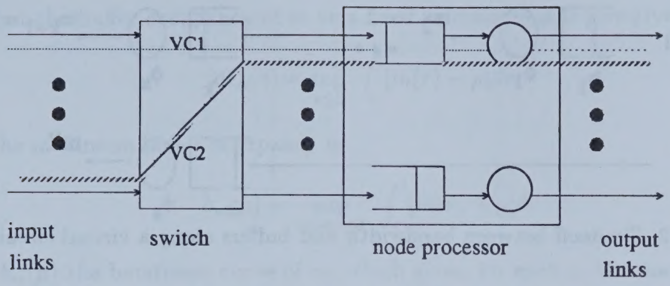


Figure 1: Description of a node

the service discipline determines how the bandwidth at the link is to be shared among these virtual circuits. Since each output link has its own buffer and bandwidth resources, the service discipline for each link is specified and analyzed independently.

The traffic on a virtual circuit over an input link is modeled as some message m. The traffic on the same virtual circuit leaving the node over an output link is similarly modeled as another message m'. Fix attention on a particular output link and suppose n virtual circuits occupy this link. Then the service discipline of that link can be abstractly defined as a mapping Φ,

$$\Phi(m_1, ..., m_n) \;=\; (m_1', ..., m_n'). \tag{1}$$

We assume that the switch has a sufficiently large capacity. The only scarce resources are the buffers and bandwidth of each output link. The node processor is then effectively described by a collection of service discipline maps, Φ, one for each output link.

In this paper we are concerned with fixed rate servers which provide guaranteed bandwidth to each virtual circuit in the presence of other connections. We may thus analyzed each virtual circuit independently; so $n = 1$ in (1) and the output link of a node involved in this connection is defined as a mapping Φ from the input message m to its output message $m' = \Phi(m)$. We define Φ by the service *allocation* $\phi(t)$ which allocate $\phi(t)$ cps to m at time t. The buffer occupancy at time t is

$$b_m(t) = \sup_{s \le t} \int_s^t [m(r) - \phi(r)]dr.$$

The output message $m' = \Phi(m)$ is then given by

$$m'(t) = \begin{cases} \phi(t) & \text{if } b_m(t) > 0 \\ \min\{\phi(t), m(t)\} & \text{if } b_m(t) = 0 \end{cases}.$$

We will henceforth use ϕ and Φ interchangeably.

A virtual circuit is defined by a path consisting of the output links $\Phi_1, ..., \Phi_k$ of several nodes. Its behavior is given by the compoiste mapping $\Phi = \Phi_k \circ ... \circ \Phi_1$ from the input message m to its output message $m' = \Phi(m)$,

$$m_1 := m;\; m_{i+1} = \Phi_i(m_i), 1 \le i < k;\; m' := m_k.$$

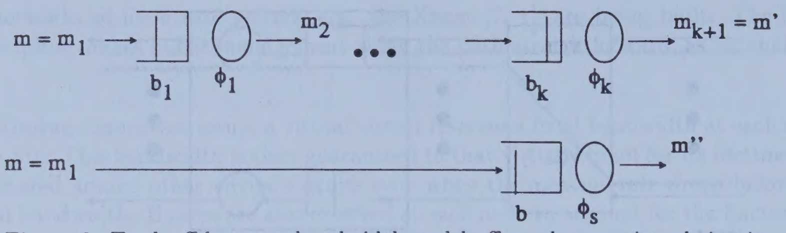


Figure 2: Tradeoff between bandwidth and buffers along a virtual circuit

A fixed rate server $\phi(t) \equiv \lambda$ allocates a bandwidth of λ cps to its input message. The following theorem expresses the behavior of a connection spanning several nodes in terms of that of a single fixed rate server and illustrates the connection between bandwidth and buffers along a virtual circuit.

Theorem 1 *Consider the virtual circuits in Figure 2. Suppose* $\phi_i \equiv \mu_i$, $i = 1, \cdots, k$, *and* $\phi_s \equiv \mu_s$ *where* $\mu_s := \min\{\mu_i \mid i = 1, \cdots, k\}$. *Then, for all* m,

1. $\sum b_i(t) \equiv b(t)$
2. $m'(t) \equiv m''(t)$.

Hence we can potentially control the buffering requirement along the path by appropriately choosing the bandwidth allocation μ_i's at the nodes.

3 Bandwidth and Buffer Allocation

In this section we consider the optimal bandwidth and buffer allocation in a network of fixed rate servers. We assume source routing in which the route selection is independent of bandwidth allocation in the network. We will first review the concept of burstiness curve, our characterization of a message, which suggests a buffer reservation scheme in the network. We then define optimality and show that the optimal bandwidth allocation problem is equivalent to a two-player nonzero sum game. The solution of the game gives an optimal bandwidth and buffer allocation, and the probability a connection request will be rejected. The formulation suggests a practical implementation scheme.

3.1 Buffer Reservation

We do not try to capture burstiness of a message by a single number such as the ratio of peak to average rate . Instead, the burstiness of m measures its buffering requirement.

Recall that the buffer occupancy of m at a fixed rate server $\phi \equiv \mu$ is given by

$$b_m(t) = \sup_{s \leq t} \int_s^t [m(r) - \mu] dr$$

and so the maximum buffer occupancy is

$$b_m(\mu) = \sup_{0 \leq s \leq t \leq T} \int_s^t [m(r) - \mu] dr \tag{2}$$

We call $b_m(\mu)$ the burstiness curve of m, which gives, for each μ, the maximum number of cells need to be buffered when m is served at a maximum rate of μ. We will drop the subscript and write b instead of b_m when there is no danger of confusion.

The burstiness curves introduce a partial order on the message space [8]. We say m_1 (or b_1) is less bursty than m_2 (or b_2) if

$$b_1(0) = b_2(0) \quad \text{and} \quad b_1(\mu) \leq b_2(\mu),\ \mu \geq 0$$

and we say a server ϕ is burst reducing if for all input message m, $\Phi(m)$ is less bursty than m. It is shown in [10] that fixed rate servers are burst reducing.

Consider a virtual circuit spanning nodes $\phi_1, \cdots, \phi_k$. Theorem 1 asserts that the total number of cells stored in the connection is equal to that stored in a hypothetical connection consisting of a single bottleneck node with rate μ_s. Hence the excess cells that need to be buffered along the entire path at any time t is upper bounded by $b(\mu_s)$. However, since output links at different nodes cannot share buffers, a connection needs to reserve a total number of buffers greater or equal to $b(\mu_s)$. We can estimate the buffering requirement at each node along the path using the burst reducing property of fixed rate servers. Since the intermediate messages are less bursty than the input message, if the connection reserves a bandwidth of μ_i at output link i along its path, it needs only $b(\mu_i)$ buffers at node i, where b is the burstiness curve of the *input* message. We note here that this buffer reservation scheme is simple but conservative as an intermediate node with rate μ_i may need strictly less than $b(\mu_i)$ buffers.

3.2 Optimal Bandwidth and Buffer Allocation

Suppose the network consists of I output links, labelled $i = 1, \cdots, I$; see figure 3. Link i has a bandwidth capacity of C_i cps and a storage capacity of B_i buffers. Let R denotes the finite set of routes. A route $r \in R$ spanning the output links $l_1, \cdots, l_k$ is denoted by $r = \{l_1, \cdots, l_k\}$. Messages requesting route $r \in R$ arrive according to a Poisson process of rate λ_r and have the same burstiness curve b_r. As r varies it indexes independent Poisson streams. To each message admitted on route r is allocated μ_{ir} amount of bandwidth and $b_r(\mu_{ir})$ amount of storage at output link $i \in r$. A request on route r is lost if at any output link $i \in r$ there is less than μ_{ir} spare bandwidth *or* less than $b_r(\mu_{ir})$ spare storage. Otherwise the request is accepted and holds μ_{ir} amount of bandwidth and $b_r(\mu_{ir})$ amount of storage at each link $i \in r$. These resources are simultaneously released at the end of the holding time. The holding time of the virtual circuits on route r is generally distributed with mean $T_r < \infty$.

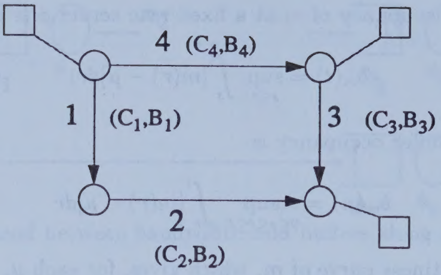


Figure 3: A network of fixed rate servers

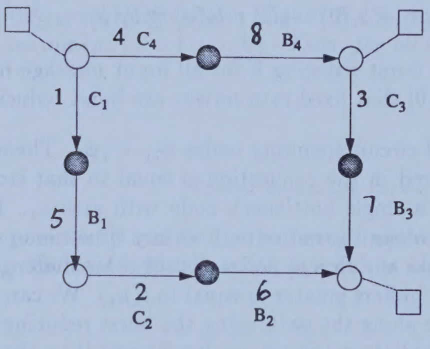


Figure 4: Equivalent circuit-switched network

Suppose each connection request on route $r \in R$ worths w_r. Let $A_r(\mu)$, $\mu = (\mu_{ir}, i \in r, r \in R)$, denote the admission probability, the probability a connection request on route r is admitted, under the bandwidth allocation μ. Let $\nu_r := \lambda_r T_r$ denote the average number of connection requests per mean holding time T_r. Then it is desired to allocate the bandwidth μ_{ir} and buffers $b_r(\mu_{ir}), i \in r, r \in R$, so as to maximize the rate

$$\sum_r w_r \nu_r A_r(\mu)$$

the network earns its revenue.

To this end, we reduce our ATM model to an equivalent model of circuit-switched networks studied in [11, 12, 13]. See figure 4. Replace each output link i of capacities (C_i, B_i) in the ATM model by two hypothetical links i and $I+i$ in tandem, link i of capacity C_i and link $I+i$ of capacity B_i. Each route r that contains output link i in the ATM model corresponds to a route r that contains links i and $I+i$ in the equivalent model. Route requests and message durations are identical for the two models. Furthermore, to each message admitted on r in the equivalent model is allocated μ_{ir} resource at link i and $b_r(\mu_{ir})$ resource at link $I+i$, $i \in r$. A message is lost, in the equivalent model, if at any link $i \in r$ there is insufficient spare capacity. An accepted request on r simultaneously holds μ_{ir} and $b_r(\mu_{ir})$ at links i and $I+i$, $i \in r$ for a duration generally distributed with mean T_r. The admission probability in the equivalent model is shown in [11] to approach, asymptotically as both the capacities (C_i, B_i) and the request rates ν_r increase,

$$A_r(\mu) \;=\; exp[-\sum_{i\in r}(\alpha_i\mu_{ir} + \beta_i b_r(\mu_{ir}))]$$

where $(\alpha, \beta) := (\alpha_1, \cdots, \alpha_I, \beta_1, \cdots, \beta_I)$ solves the following minimization

$$\min_{(\alpha,\beta)\geq 0} \quad \sum_{r\in R}\nu_r exp[-\sum_{i\in I}(\alpha_i\mu_{ir} + \beta_i b_r(\mu_{ir}))] + \sum_{i\in I}(\alpha_i C_i + \beta_i B_i). \tag{3}$$

Suppose messages on route r have a minimum bandwidth requirement specified by $\underline{\mu}_{ir}$ at link $i, i \in r$. Then the allocation μ_{ir} has to satisfy $\underline{\mu}_{ir} \leq \mu_{ir} \leq M_r$, where we assume $M_r < C_i, i \in r$, M_r being the peak rate of messages on route r. Define the vectors $\mu_r, b_r(\mu_r) \in \Re^I$ by

$$\mu_r(i) \;:=\; \begin{cases} \mu_{ir} & \text{if } i \in r \\ 0 & \text{otherwise} \end{cases}$$

$$b_r(\mu_r)(i) \;:=\; \begin{cases} b_r(\mu_{ir}) & \text{if } i \in r \\ 0 & \text{otherwise} \end{cases}$$

and similarly define $\underline{\mu}_r, C_r, B_r \in \Re^I$. Let μ be the concatenation of $\mu_r, r \in R$.

Definition 1 *We say a bandwidth allocation μ optimal if it maximizes the asymptotic revenue rate $R(\mu)$, i.e., it solves*

$$\max_{\mu\geq\underline{\mu}} \quad R(\mu) := \sum_r w_r\nu_r A_r(\mu) \tag{4}$$

Proposition 1 *Consider the following two-player nonzero sum game*

$$\begin{aligned} &\textit{player 1:} \quad \min\nolimits_{\mu\geq\underline{\mu}} \quad F(\mu,\alpha,\beta) && (5)\\ &\textit{player 2:} \quad \min\nolimits_{(\alpha,\beta)\geq 0} \quad G(\mu,\alpha,\beta) && (6)\end{aligned}$$

where

$$\begin{aligned} F(\mu,\alpha,\beta) \;&:=\; -\sum_r w_r\nu_r f_r(\mu_r,\alpha,\beta) \\ G(\mu,\alpha,\beta) \;&:=\; \sum_r \nu_r f_r(\mu_r,\alpha,\beta) + <\alpha, C> + <\beta, B> \\ f_r(\mu,\alpha,\beta) \;&:=\; exp(-<\alpha,\mu_r> - <\beta, b_r(\mu_r)>) \end{aligned}$$

and $<\cdot,\cdot>$ denotes inner product. The Stackelberg strategy μ^* for the leader, player 1, is an optimal allocation for (4).

The solution of (4) is assured by the following theorem.

Theorem 2 *There exists a Stackelberg solution* $(\mu^*, \alpha^*, \beta^*)$ *for the game (5) and (6) such that*

$$\begin{aligned} R(\mu^*) &= \max_{\mu \geq \underline{\mu}} R(\mu) \\ A_r(\mu^*) &= f_r(\mu^*_r, \alpha^*, \beta^*). \end{aligned}$$

The problem (4) admits an even simpler structure when connections on different routes worth the same.

Theorem 3 *When* $w_r = 1, r \in R$, *the bandwidth allocation problem (4) is equivalent to the following zero sum game*

$$\max_{\mu \geq \underline{\mu}} \ \min_{(\alpha,\beta)\geq 0} \quad G(\mu, \alpha, \beta).$$

In that case, there exists a saddle point $(\mu^*, \alpha^*, \beta^*)$ *with*

$$\max_{\mu \geq \underline{\mu}} \ \min_{(\alpha,\beta)\geq 0} \quad G(\mu, \alpha, \beta) = G(\mu^*, \alpha^*, \beta^*) = \min_{(\alpha,\beta)\geq 0} \ \max_{\mu \geq \underline{\mu}} \quad G(\mu, \alpha, \beta)$$

at which

$$\begin{aligned} R(\mu^*) &= \max_{\mu \geq \underline{\mu}} R(\mu) \\ A_r(\mu^*) &= f_r(\mu^*_r, \alpha^*, \beta^*) \end{aligned}$$

Under allocation μ, the network produces services, or outputs, represented by $y(\mu) := (\nu_r A_r(\mu), r \in R)$, in which each component $\nu_r A_r(\mu)$ worths w_r. The sales price $w_r = (w_r, r \in R)$ yields an optimal output y^* which in turn determines an optimal bandwidth allocation μ^*. The optimal allocation μ^* induces a fictitious resource price (α^*, β^*), which we discuss below.

3.3 Implementation

We assume throughout this section that $w_r = 1, r \in R$. Suppose the burstiness curve of messages on route r is only known to individual messages but not to the network nor to messages on other routes. Moreover, suppose it may vary slowly (much slower than the mean holding time T_r) and it is desired to adaptively adjust the optimal allocation. Without $b_r, r \in R$, the network cannot solve (4). In Theorem 3, player 1 tries to maximize $G(\mu, \alpha, \beta)$ over $\mu \geq \underline{\mu}$ which is equivalent to

$$\min_{\mu \geq \underline{\mu}} \quad \sum_{i \in r} (\alpha_i \mu_{ir} + \beta_i b_r(\mu_{ir}))$$

By the theorem if player 2 chooses the optimal (α^*, β^*), then player 1 will choose the optimal μ^*. In that case optimal μ^* for player 1 satisfies

$$\begin{aligned} \mu^*_{ir} &= \underline{\mu}_{ir} && \text{if } \alpha^*_i > -\beta^*_i \frac{d}{d\mu} b_r(\underline{\mu}) \\ \mu^*_{ir} &= M_r && \text{if } \alpha^*_i < -\beta^*_i \frac{d}{d\mu} b_r(M_r) \\ \alpha^*_i &= -\beta^*_i \frac{d}{d\mu} b_r(\mu^*_{ir}) && \text{otherwise.} \end{aligned}$$

The decentralized nature of the optimal allocation is striking: given (α^*, β^*), the optimal bandwidth and buffer allocation $(\mu_{ir}^*, b_r(\mu_{ir}^*))$ at link i on route r depends only on parameters pertaining to link i. This suggests the following control scheme.

Suppose each source desination pair is given some tokens every month or year for purchasing bandwidth and buffers during connection setup. The network charges each connection, in the fictitious currency, per unit time, α_i per unit of bandwidth and β_i per unit of storage at link i. The expected connection cost on route r is therefore

$$T_r \sum_{i \in r} (\alpha_i \mu_{ir} + \beta_i b_r(\mu_{ir}))$$

and hence if the network charges according to (α^*, β^*), a connection, in order to minimze its expected connection cost, will indeed reserve the optimal μ_r^*. Suppose time is divided into update periods $n = 0, 1, \cdots$. The network posts a price, in fictitious currency, (α^n, β^n) at time n and observes the requested bandwidth and buffer allocation (μ^n, b^n) in the entire network in that period. Then it solves (3) with allocation (μ^n, b^n) and uses its minimizer $(\alpha^{n+1}, \beta^{n+1})$ to be the resource price for the next period, and the cycle repeats. The following theorem assures the convergence to optimal allocation.

Theorem 4 *Given (α^0, β^0), construct the sequence $(\mu^n, \alpha^{n+1}, \beta^{n+1}), n \geq 0$, by*

$$\begin{aligned} \alpha_i^n \mu_{ir}^n + \beta_i^n b_r(\mu_{ir}^n) &= \min_{\mu_{ir} \geq \underline{\mu}_{ir}} \quad \alpha_i^n \mu_{ir} + \beta_i^n b_r(\mu_{ir}), \quad i \in r, \ r \in R \\ G(\mu^n, \alpha^{n+1}, \beta^{n+1}) &= \min_{(\alpha,\beta) \geq 0} \quad G(\mu^n, \alpha, \beta). \end{aligned} \tag{7}$$

Then $(\mu^n, \alpha^{n+1}, \beta^{n+1})$ converges to some $(\mu^*, \alpha^*, \beta^*)$ that satisfies the conclusion of Theorem 3.

Note that (7) involes $2I$ variables as opposed to $R \times I$ variables for (4), where R is typically much larger than I.

Acknowledgement

Part of this work is done while the first author is an intern at Apple Computer, Inc. during the summer of 1991. The support of K.M. Nichols and M. MacDougall is gratefully acknowledged.

References

[1] C. Kalmanek, H. Kanakia, and S. Keshav, "Rate controlled servers for very high speed networks," *Proceedings of Globecom 1990*, December 1990.

[2] L. Zhang, *A new architecture for packet switching network protocols.* PhD thesis, MIT, 1989.

[3] D. C. Verma, H. Zhang, and D. Ferrari, "Delay jitter control for real-time communication in a packet switching network," *Proceedings of TriComm'91*, March 1991.

[4] H. Zhang and S. Keshav, "Comparison of rate-based service disciplines," *to appear in SIGCOMM'91*, Sept. 1991.

[5] D. Ferrari and D. Verma, "A scheme for real-time channel establishment in wide-area networks," *IEEE Journal on Selected Areas in Communications*, vol. 8, April 1990.

[6] D. Ferrari, "Client requirements for real-time communication services," *IEEE Communications Magazine*, November 1990.

[7] C. R. Kalmanek, "Xunet 2: A nationwide testbed in high-speed networking," *CS TR, AT&T Bell Labs*, March 1991.

[8] S. Low and P. Varaiya, "A simple theory of traffic and resource allocation in ATM," *Accepted by Globecom'91*, December 1991.

[9] J. Hui, *Switching and traffic theory for integrated broadband networks.* Kluwer Academic Publishers, 1990.

[10] S. Low and P. Varaiya, "A traffic theory and its application to resource allocation in ATM." To be submitted for publication.

[11] F. P. Kelly, "Blocking probabilities in large circuit-switched networks," *Adv. Appl. Prob.*, no. 18, 1986.

[12] F. P. Kelly, "Blocking and routing in circuit-switched networks," *Teletraffic Analysis and Computer Performance Evaluation*, 1986.

[13] F. P. Kelly, "Routing in circuit-switched networks: optimization, shadow prices and decentralization," *Adv. Appl. Prob.*, no. 20, 1988.

Input Buffer Requirements For Round Robin Polling Systems With Nonzero Switchover times

Galen Sasaki*
Department of Electrical and Computer Engineering
The University of Texas, Austin, TX 78712-1084

Abstract

We consider a packet switch network that can be modeled as a round robin polling system with nonzero switchover times. The input traffic is assumed to obey the (σ, ρ) regularity conditions recently proposed by Cruz. For the exhaustive, partially-gated, and gated services, we provide numerical methods to compute upper bounds for the size of packet buffers to insure no packets are lost.

1. Introduction

The system considered in this paper is a packet switch that can be modeled as a polling system. The packet switch consists of N nodes labeled $0, 1, ..., N-1$, and a high-capacity broadcast link with finite transmission rate μ. Each node has an *input link* through which incoming packets arrive. The transmission capacity of each input link is λ, which has a positive and possibly infinite value. If a packet is transmitted on a link then it is transmitted continuously using the full capacity of the link. Each node has a buffer to store the incoming packets. Packets leave the buffer and node through the high-capacity broadcast link, and the broadcast link serves the nodes in round robin order according to node labels. Initially at time 0 the system is empty of bits. Packets have nonzero length, can be arbitrarily small, and have maximum size P.

For each node i and time t, let $R_i(t)$ equal the rate at which data is flowing on the input link, $P_i(t)$ equal the number of bits buffered, and $\Lambda_i(t)$ equal the number of bits buffered corresponding to a packet not completely received on the input link. If $\lambda = \infty$ then $\Lambda_i(t) = 0$ for all t. Our model allows packet lengths to be noninteger, which is reasonable if the maximum packet length is long. Note that $R_i(t) \in \{0, \lambda\}$ if λ is finite, otherwise $R_i(t)$ is the sum of delta functions. The delta function used in this paper is the function $\delta(t)$. It has the property that for any $t' \leq t$, $\int_{t'}^{t} \delta(\tau)d\tau = I_{[0 \in (t', t]]}$, where $I_{[\,]}$ is the indicator function (i.e., $I_{[A]} = 1$, if A is true; and $I_{[A]} = 0$, if A is false.) Call the set $\{R_i(t) : 0 \leq i \leq N-1\}$ the *input traffic*, and assume that

$$\int_{x}^{y} R_i(t)dt \leq \sigma + \rho(y - x) \quad \text{for any } x \leq y \tag{1.1}$$

---

* Research supported by the National Science Foundation under grants NCR-8809410 and NCR-8958556 and an equipment donation from SUN Microsystems, Inc.

where σ and ρ are nonnegative constants. This characterization of packet flow was proposed by Cruz [1]. The value of ρ is an upper bound on the steady-state rate of packet flow and σ is a parameter characterizing the *burstiness* of $R_i(t)$.

Let M_i be the set of times that the broadcast link serves node i. For all nodes i and times t', t such that $t' \leq t$, $P_i(t') + \int_{t'}^{t} R_i(\tau) - \mu I_{[\tau \in M_i]} \, d\tau = P_i(t) \geq 0$. The last equality implies that bits transmitted by the broadcast link are not stored.

The system is *packet integral*, which means that the broadcast link will transmit a packet only after it has been completely received and will transmit it without interruption. The time delay that the broadcast link takes to switch from node $(i-1) \bmod N$ to the subsequent node i is constant and denoted by v_i. The time interval for this delay is called a *polling interval* for node i during which the broadcast link does not transmit. Packets from node i are transmitted by the broadcast link during single *data intervals* that immediately follow polling intervals for node i. For notational simplicity, let $V = \sum_{i=0}^{N-1} v_i$, $\beta = N\rho/\mu$, $D = \sigma + \rho P/\lambda$. A node is said to be *busy* if the broadcast link is transmitting a packet in it. The system is said to be *busy* if one of the nodes is busy. The system (or node) is called *idle* if it is not busy.

We consider the *exhaustive*, *partially gated*, and *gated* service policies. For exhaustive service, the broadcast link will continually transmit packets at a node until there are no complete packets. For partially gated service (resp., gated service), the broadcast link will transmit all packets at a node that are completely received at the end (resp., start) of the most recent polling interval for the node.

We make the following assumptions about the packet switch system. First, node 0 is polled first. Second, $V > 0$. Third, for all nodes i and times $t \leq 3V$, $R_i(t) = 0$. Then the broadcast link will poll each node at least three times before data arrives. Fourth, we assume that $v_i = V/N$ for all nodes i if the service policy is gated. Finally, we assume that $\beta \in (0, 1)$.

Let P^* equal the supremum of $\max_{0 \leq i < N} P_i(t)$ over all $t \geq 0$ and input traffic that satisfies (1.1). Note that P^* is an upper bound on the size of node buffers to insure no packet loss. In Section 2, we define an optimization problem that has a solution that is an upper bound for P^* for the three service policies. In Section 3, we define a set of equations whose solution is an upper bound for $\limsup_N P^*$. Final remarks are left in Section 4.

We conclude the introduction by citing related work. The system considered in this paper is studied in [2] and [3] but for $V = 0$ and $\sigma = 0$. More efficient service policies are considered in [4][5]. Similar traffic models applied to manufacturing systems appear in [6][7][8].

2. An Optimization Problem for Computing Storage Requirements

In this section, we provide an optimization problem whose solution is an upper bound for P^* for exhaustive, partially gated, and gated service policies. The technique of converting the problem of finding a bound for P^* into an optimization problem was done first in [1]. However, we consider a more general system, and our optimization problem has additional constraints that lead to better solutions.

We proceed through preliminary definitions and lemmas before the main result of this section, Theorem 3.1.

Definitions. Let $s_i(j)$ and $t_i(j)$ be the start and end times of the jth data interval for node i. Let $\Delta_i(j) = t_i(j) - s_i(j)$, the duration of the data interval, and let $W_i(j) = \sum_{k=i}^{N-1} \Delta_k(j) + \sum_{k=0}^{i-1} \Delta_k(j+1)$, which is the amount of time the broadcast link is busy during interval $(s_i(j), s_i(j+1)]$.

Lemma 2.0. For all $t \geq t' \geq 0$, $\Lambda_i(t') + \int_{t'}^{t} R_i(x)dx \leq D + \rho(t-t')$.

Proof. Note that $\Lambda_i(t') + \int_{t'}^{t} R_i(x)dx = \int_{t'-\Lambda_i(t')/\lambda}^{t} R_i(x)dx \leq \int_{t'-P/\lambda}^{t} R_i(x)dx \leq \sigma + \rho(t - t' + P/\lambda)$, where the last inequality follows from (1.1). Then the lemma is implied because the $D = \sigma + \rho P/\lambda$. □

Lemma 2.1. For any $j \geq 0$ and node i, $\Delta_i(j) \leq (P^* + \sigma)/(\mu - \rho)$. If the service policy is partially gated or gated then $\Delta_i(j) \leq P^*/\mu$.

Proof. Note that $P_i(s_i(j)) + \int_{s_i(j)}^{t_i(j)} R_i(x)dx - \mu\Delta_i(j) = P_i(t_i(j)) \geq 0$. Applying (1.1) to the last inequality we have $P_i(s_i(j)) + \sigma + \rho\Delta_i(j) - \mu\Delta_i(j) \geq 0$. Then $\Delta_i(j) \leq \frac{P_i(s_i(j))+\sigma}{\mu-\rho} \leq \frac{P_i^*+\sigma}{\mu-\rho}$. Next note that for partially gated and gated policies, $\Delta_i(j) \leq \frac{P_i(s_i(j))}{\mu} \leq \frac{P^*}{\mu}$. □

Next we focus on a bound for $P_0(t)$. Let $\Gamma_i(j) = t_i(j)$ for exhaustive service; let $\Gamma_i(j) = s_i(j)$ for partially gated service; and let $\Gamma_i(j) = s_i(j) - v_i$ for gated service. Note that packets received at node i during the interval $(\Gamma_i(j-1), \Gamma_i(j)]$ will be transmitted during the jth data interval of node i, i.e., $[s_i(j), t_i(j)]$.

Lemma 2.2. Let B be an upper bound for P^*. Let t be a time when node 0 is busy. Let J be such that $t \in [s_0(J+1), t_0(J+1)]$. Let J' be the largest integer such that $J' < J$ and $W_0(J') \leq \frac{\beta V}{1-\beta}$. ($J'$ exists because the system is empty during its first three polling cycles.)

Let $\Delta_i = \Delta_i(J)$ and $\Delta'_i = \Delta_i(J')$. Let γ, τ, and τ'_i be defined as follows. For exhaustive service, let $\gamma = (B+\sigma)/(\mu-\rho)$, $\tau = 0$, and $\tau'_i = 0$. For partially gated service, let $\gamma = B/\mu$, $\tau = 0$, and $\tau'_i = \Delta'_i$. For gated service, let $\gamma = B/\mu$, $\tau = V/N$, and $\tau'_i = \Delta'_i + V/N$. Let

$$K_i = H_i + \rho \sum_{j=i+1}^{N-1} \Delta'_j + \sum_{k=J'+1}^{J-1} [\rho(W_0(k) + V) - \mu\Delta_i(k)] + \rho V \tag{2.6}$$

$$H_i = \min\{P_i(s_i(J')) + \sigma - (\mu-\rho)\Delta'_i, \quad D + \rho\tau'_i\} \tag{2.7}$$

Then the following inequalities hold for exhaustive, gated, and partially gated service policies: $P_0(t) \leq D_0 + \rho_0[V + V/N + \sum_{i=0}^{N-1} \Delta_i]$ (Ineq. 2.8); $H_i \leq B + \sigma - (\mu - \rho)\Delta_i'$ (Ineq. 2.9); $H_i \leq D + \rho\tau_i'$ (Ineq. 2.10); $\sum_{i=0}^{N-1} \Delta_i' \leq \beta V/(1-\beta)$ (Ineq. 2.11); $\Delta_i, \Delta_i' \leq \gamma$ (Ineq. 2.12); $K_i \geq 0$ (Ineq. 2.13);

$$\sum_{i=0}^{N-1} K_i \leq \sum_{i=0}^{N-1} H_i + \sum_{i=0}^{N-1} \Delta_i' i\rho + \beta\mu V \tag{2.14}$$

$$\Delta_i \leq \frac{K_i}{\mu - \rho} + \frac{\rho}{\mu - \rho} \sum_{k=0}^{i-1} \Delta_k \tag{2.15}$$

Proof. We shall prove inequalities (2.8) through (2.12), (2.14). (2.15), and (2.13) in this order. First note that $P_0(t) = \int_{\Gamma_0(J)}^{t} R_0(x)dx + \Lambda_0(\Gamma_0(J)) - \mu(t - s_0(J+1)) \leq D + \rho[t - \Gamma_0(J)] - \mu(t - s_0(J+1))$, where the last inequality is due to Lemma 2.0. Then $P_0(t) \leq D + \rho[s_0(J+1) - \Gamma_0(J)] - (\mu - \rho)(t - s_0(J+1)) \leq D + \rho[s_0(J+1) - \Gamma_0(J)] \leq D + \rho[V + V/N + \sum_{i=0}^{N-1} \Delta_i(J)]$, which implies (2.8).

Inequalities (2.9) and (2.10) are immediate from the definition of H_i, Inequality (2.11) follows from our choice of J', and Inequality (2.12) follows from Lemma 2.1 and $B \geq P^*$.

If we sum the left and right hand sides of (2.6), we get

$$\begin{aligned}
\sum_{i=0}^{N-1} K_i &= \sum_{i=0}^{N-1} H_i + \sum_{i=0}^{N-1} \rho \sum_{j=i+1}^{N-1} \Delta_j(J') \\
&\quad + \sum_{k=J'+1}^{J-1} \left[\sum_{i=0}^{N-1} \rho(W_0(k) + V) - \mu W_0(k) \right] + \sum_{i=0}^{N-1} \rho V \\
&= \sum_{i=0}^{N-1} H_i + \sum_{i=0}^{N-1} \Delta_i(J') \sum_{k=0}^{i-1} \rho + \mu \sum_{k=J'+1}^{J-1} [\beta V - (1-\beta)W_0(k)] + \beta\mu V
\end{aligned}$$

The last summation is at most zero because $\frac{\beta V}{1-\beta} < W_0(k)$ for $k = J'+1, J'+2, ..., J-1$. Hence, (2.14) follows.

We now turn to (2.15) and (2.13). Let t' be either $s_0(J)$ or $t_i(J)$. Let $\overline{\Delta} = 0$, if $t' = s_0(J)$; and let $\overline{\Delta} = \Delta_i(J)$, if $t' = t_i(J)$. Observe that

$$P_i(s_i(J')) + \int_{s_i(J')}^{t'} R_i(x)dx - \mu \sum_{k=J'}^{J-1} \Delta_i(k) - \mu\overline{\Delta} = P_i(t') \geq 0 \tag{2.17}$$

$$P_i(t_i(J')) + \int_{t_i(J')}^{t'} R_i(x)dx - \mu \sum_{k=J'+1}^{J-1} \Delta_i(k) - \mu\overline{\Delta} = P_i(t') \geq 0 \tag{2.18}$$

Applying Inequality (1.1) to (2.17) we have,

$$P_i(s_i(J')) + \sigma + \rho[t' - s_i(J')] - \mu \sum_{k=J'}^{J-1} \Delta_i(k) - \mu\overline{\Delta} \geq 0 \tag{2.19}$$

We turn to Inequality (2.18). Note that

$$\begin{aligned} P_i(t_i(J')) + \int_{t_i(J')}^{t'} R_i(x)dx &= \Lambda_i(\Gamma_i(J')) + \int_{\Gamma_i(J')}^{t_i(J')} R_i(x)dx + \int_{t_i(J')}^{t'} R_i(x)dx; \\ &= \Lambda_i(\Gamma_i(J')) + \int_{\Gamma_i(J')}^{t'} R_i(x)dx; \\ &\leq D + \rho(t' - \Gamma_i(J')) \end{aligned} \tag{2.20}$$

where (2.20) follows from Lemma 2.0. Combining (2.18) and (2.20), we have

$$D + \rho\tau_i(J') + \rho[t' - t_i(J')] - \mu \sum_{k=J'+1}^{J-1} \Delta_i(k) - \mu\overline{\Delta} \geq 0 \tag{2.21}$$

Equalities (2.19) and (2.21) imply.

$$H_i + \rho[t' - t_i(J')] - \mu \sum_{k=J'+1}^{J-1} \Delta_i(k) - \mu\overline{\Delta} \geq 0 \tag{2.22}$$

Let $n = -1$, if $t' = s_0(J)$; and let $n = i$, if $t' = t_i(J)$. Then

$$t' - t_i(J') \leq V + \sum_{k=i+1}^{N-1} \Delta_i(J') + \sum_{k=J'+1}^{J-1} (V + W_0(k)) + \sum_{k=0}^{n} \Delta_i(J) \tag{2.23}$$

Note that (2.22) and (2.23) imply

$$H_i + \rho \sum_{k=i+1}^{N-1} \Delta_k(J') + \sum_{k=J'+1}^{J-1} [\rho\,(W_0(k) + V) - \mu\Delta_i(k)] + \rho V$$

$$-\mu\overline{\Delta} + \rho \sum_{k=0}^{n} \Delta_k(J) \geq 0$$

Then $K_i - \mu\overline{\Delta} + \rho_i \sum_{k=0}^{n} \Delta_k(J) \geq 0$. Note that the last inequality implies (2.15) and (2.13) if $t' = t_i(J)$ and $t' = s_0(J)$, respectively. □

Theorem 2.1. Consider the following linear programming problem that has the variables $\{\Delta_i, \Delta_i', K_i, H_i \,:\, 0 \leq i \leq N-1\}$ and B. *Problem LP*: Maximize $D + \rho[V + V/N + \sum_{n=0}^{N-1} \Delta_n]$ subject to the variables satisfying Inequalities (2.8) – (2.15) in Lemma 2.2. and the inequality $D + \rho[V + V/N + \sum_{n=0}^{N-1} \Delta_n] \geq B$. The maximum of LP is an upper bound for P^*.

Proof. Suppose we modify the problem LP so that the inequality $D+ \rho[V + V/N+ \sum_{n=0}^{N-1} \Delta_n] \geq B$ is replaced by $B = P^*$. Let B^1 be the maximum of this modified problem. From Lemma 2.2., we know that B^1 is an upper bound for $P_0(t)$ for any t while node 0 is busy and any input traffic satisfying (1.1). Note that $P_0(t)$ is nondecreasing during idle periods for node 0. Hence, $P_0(t)$ is maximized at the end of idle periods, which is the beginning of a busy period. We can conclude that B^1 is an upper bound for $P_0(t)$ for any time t and input traffic satisfying (1.1). In fact, due to the symmetry of the system and the parameters chosen, B^1 is an upper bound for $P_i(t)$ for any t and any input traffic satisfying (1.1). Hence, $B^1 \geq P^*$. This implies there are values for the variables B and $\{\Delta_i, \Delta_i', K_i,\ H_i : 0 \leq i \leq N-1\}$ that satisfy the constraints of Problem LP and the resulting objective function value is at least P^*. The theorem is implied. □

3. Asymptotic Upper Bounds

In this section we evaluate the asymptotic solution of Problem LP as N goes to infinity. We proceed by converting the problem into two subproblems then characterizing the solutions of the subproblems. Consider Problem LP^*: Maximize $\sum_{i=0}^{N-1} H_i + \sum_{i=0}^{N-1} \Delta_i' i\rho + \beta\mu V$, subject to $\sum_{i=0}^{N-1} \Delta_i' \leq \frac{\beta V}{1-\beta}$ and, for each i, $0 \leq \Delta_i' \leq \gamma$, $H_i \leq D + \rho\tau_i'$, and $H_i \leq B + \sigma - (\mu - \rho)\Delta_i'$. Let $G^*(B)$ be the maximum value for LP^*. Also, consider Problem LP^{**}: Maximize $D + \rho[V + \tau + \sum_{i=0}^{N-1} \Delta_i]$, subject to $\sum_{i=0}^{N-1} K_i \leq G^*(B)$ and, for each i, $0 \leq \Delta_i \leq \gamma$, $K_i \geq 0$, and $\Delta_i \leq \frac{K_i}{\mu-\rho} + \frac{\rho}{\mu-\rho} \sum_{k=0}^{i-1} \Delta_k$. Let $G^{**}(B)$ be the maximum value for LP^{**}. Note that $G^{**}(B)$ equals the maximum for Problem LP.

Lemma 3.1. Suppose $B \geq D - \sigma + \rho\frac{V}{N}$. Let $\psi = (B + \sigma - D)/(\mu - \rho)$ for exhaustive and partially gated service; and let $\psi = (1/\mu)(B + \sigma - D - \rho V/N)$ for gated service. Let J be the smallest integer such that $0 \leq J \leq N$ and $(N-J)\psi \leq \frac{\beta V}{1-\beta}$. Let $\psi^* = \frac{\beta V}{1-\beta} - (N - J)\psi$. Then

$$G^*(B) = ND + \beta\mu V + \rho \times \begin{cases} (J-1)\psi^* + \psi\sum_{i=J}^{N-1} i; & \text{for exhaustive;} \\ J\psi^* + \psi\sum_{i=J}^{N-1} (i+1); & \text{for partially gated;} \\ J\psi^* + \psi\sum_{i=J}^{N-1} (i+1) + V; & \text{for gated;} \end{cases}$$

Proof. For notational convenience, during the proof let $X_i = D + \rho\tau_i' + i\Delta_i'\rho$ and $Y_i = B + \sigma - (\mu - (i+1)\rho)\Delta_i'$. Note that an optimal solution for LP^* would require $H_i = \min\{B + \sigma - (\mu - \rho)\Delta_i',\ D + \rho\tau_i'\}$. Therefore, the maximum of LP^* is equal to the maximum of Problem (3.1.1): Maximize $\sum_{i=0}^{N-1} \min\{Y_i, X_i\} + \beta\mu V$, subject to $\sum_{i=0}^{N-1} \Delta_i' \leq \frac{\beta V}{1-\beta}$, and for all $i \leq N-1$, $0 \leq \Delta_i' \leq \gamma$.

Suppose $\{\Delta_j'\}_{j=0}^{N-1}$ maximizes Problem (3.1.1). We will show that for each node i, $X_i \leq Y_i$. We consider two cases. The first case is when $\Delta_i' > 0$. If $X_i > Y_i$ then Δ_i' can be decreased and remain nonnegative, leaving a larger objective function for Problem

(3.1.1). This contradicts the optimality of $\{\Delta'_j\}_{j=0}^{N-1}$, so $X_i \leq Y_i$. The second case is $\Delta'_i = 0$. Then $X_i \leq Y_i$ because of the assumption $B \geq D - \sigma + \frac{\rho V}{N}$.

Since $X_i \leq Y_i$ is equivalent to $\Delta'_i \leq \psi$, the maximum for Problem (3.1.1) is the same as the maximum for Problem (3.1.2): Maximize $\sum_{i=0}^{N-1} X_i + \beta\mu V$, subject to $\sum_{i=0}^{N-1} \Delta'_i \leq \frac{\beta V}{1-\beta}$, and for all nodes i, $0 \leq \Delta'_i \leq \min\{\psi, \gamma\}$. Since $\gamma \geq \psi$, we can replace $\min\{\psi, \gamma\}$ with ψ.

The objective function for Problem (3.1.2) is linear such that if $i > j$ then the coefficient for Δ'_i is larger than the one for Δ'_j. Hence if J is the smallest integer such that $(N - J)\psi \leq \frac{\beta V}{1-\beta}$ then Problem (3.1.2) is maximized when $\Delta'_i = \psi$, if $i \geq J$; $\Delta'_{J-1} = \frac{\beta V}{1-\beta} - (N - J)\psi$, if $i = J - 1$; $\Delta'_i = 0$, if $i < J - 1$. The proof is completed by substituting the values of Δ'_i into $\sum_{i=0}^{N-1}(D + \rho\tau'_i + i\rho\Delta'_i) + \beta\mu V$. $\square$

Lemma 3.2. Consider the problem LP^{**}. Let $J' \leq N$ be the largest integer such that $\sum_{i=0}^{J'-1}(\mu - \rho(i+1))\gamma \leq G^*(B)$. An optimal solution is for all nodes i,

$$\Delta_i = \min\{\gamma, \frac{K_i}{\mu - \rho} + \frac{\rho}{\mu - \rho}\sum_{n=0}^{i-1}\Delta_n\} \tag{3.2}$$

$$K_i = \begin{cases} (\mu - \rho(i+1))\gamma, & \text{if } i < J'; \\ G^*(B) - \sum_{n=0}^{i-1} K_n; & \text{if } i = J'; \\ 0, & \text{if } i > J'; \end{cases} \tag{3.3}$$

Proof. For notational convenience, during the proof let $Z_i = \frac{K_i}{\mu-\rho} + \frac{\rho}{\mu-\rho}\sum_{n=0}^{i-1}\Delta_n$. There exists an optimal solution for LP^{**} with the property, *Property (3.2.1): there is a J such that $\gamma \leq Z_i$, if $i < J$; and $K_i = 0$ if $i > J$.* To see why this is true, suppose that no optimal solution has this property. Then for every optimal solution, we have $n < m$, where n is the smallest value such that $\gamma > Z_n$ and m is the largest value such that $K_m > 0$. Consider an optimal solution when the difference between n and m is smallest. Decrease K_m and increase K_n by an amount $\epsilon > 0$ so that $K_m \geq 0$, $Z_n \leq \gamma$, and either $K_m = 0$ or $Z_n = \gamma$. (Note that $\sum_{i=0}^{N-1} K_i$ remains the same.) To satisfy the constraints of LP^{**}, decrease Δ_m by an amount $\epsilon' \leq \frac{\epsilon}{\mu-\rho}$ so that $0 \leq \Delta_m \leq Z_m$. Then increase Δ_n by ϵ'. Note that after the increase, $\Delta_n \leq Z_n \leq \gamma$ because initially $\Delta_n \leq Z_n$. The modifications do not decrease the value of $\sum_{i=0}^{N-1}\Delta_i$, hence the new solution is optimal. In addition, the difference between n and m is smaller which is a contradiction to our choice of solution.

We will show that there is an optimal solution with the following *Property (3.2.2): there is a J' such that $\gamma = Z_i$, if $i < J'$; $\gamma \geq Z_i$, if $i = J'$; $K_i = 0$, if $i > J'$.* Suppose we have an optimal solution with Property (3.2.1). For each $i \leq J$ such that $\gamma < Z_i$, decrease the value of K_i until $K_i = 0$ or $\gamma = Z_i$. The modified solution is optimal since it still has the same value for $\sum_{i=0}^{N-1}\Delta_i$ and satisfies the constraints of LP^{**}. The modified solution satisfies *Property (3.2.2a): $K_i = 0$ if $i \leq J$; and $\gamma < Z_i$.*

We will proceed to show that the modified solution has Property B. Let J' be the largest value such that $K_{J'} > 0$. Then $K_i = 0$ if $i > J'$. Also note that $\gamma \geq Z_{J'}$ because $J' \leq J$, $K_{J'} > 0$, and Property (3.2.2a). Suppose there is an $i < J'$ such that $\gamma \neq Z_i$. Then, from Property (3.2.1), $\gamma < Z_i$. From Property (3.2.2a) we have $K_i = 0$. Then $Z_i \leq Z_{J'}$. We can conclude $\gamma < Z_i \leq Z_{J'}$, which is a contradiction. Therefore if $i < J'$ then $\gamma = Z_i$.

For notational convenience, during the proof let $Y_i = (\mu - \rho(i+1))\gamma$. We will show that an optimal solution $\{\Delta_i, K_i : i = 0, 1, ..., N-1\}$ that satisfies Property (3.2.2) also satisfies the following *Property (3.2.3): there is a* J' *such that for each* $i < N$, *(3.2) is satisfied and* $K_i = Y_i$, *if* $i < J'$ $K_{J'} \leq Y_{J'}$; $K_i = 0$, *if* $i > J'$. Let J' be the same as in Property (3.2.2). Note that the solution must satisfy (3.2). Otherwise, we could increase the value of Δ_i without violating constraints and increase the objective function for LP^{**}. Also note that for each $i < J'$, $\Delta_i = \gamma$ due to Property (3.2.2) and (3.2). Thus, for each $i \leq J'$, $Z_i <$ (resp., $=$, $>$) γ if and only if $K_i <$ (resp., $=$, $>$) Y_i. Hence, Property (3.2.2) implies Property (3.2.3).

We now complete the proof. Let $S^j = \{\Delta_i^j, K_i^j\}_{i=0}^{N-1}$, where S^1 is an optimal solution that satisfies Property (3.2.3), and S^2 is the solution described in the Lemma. Note that for each $i < N$, $Y_i \geq 0$, Therefore, S^2 satisfies the constraints of LP^{**} and for each $i < N$, $K_i^1 \leq K_i^2$. Then $S^* = \{\Delta_i^1, K_i^2\}_{i=0}^{N-1}$ satisfies the constraints of LP^{**} and is optimal. Note that S^* must satisfy (3.2), otherwise Δ_i^1 can be increased without violating constraints and increase the objective function. Hence, $S^2 = S^*$.

□

Lemma 3.3. Suppose B, ρ, V, σ, and β are constant with N. Assume $B > P$ and that ρ, $\beta > 0$. Let $G_n^*(B)$ and $G_n^{**}(B)$ equal $G^*(B)$ and $G^{**}(B)$ when $N = n$. Then $g^*(B) = \lim_n G_n^*(B)/n$ and $G_\infty^{**}(B) = \lim_n G_n^{**}(B)/n$ exist and the following are true

$$g^*(B) = D + \rho(j_\infty \psi_\infty^* + V) + \frac{\beta\alpha}{2}(1 - (j_\infty)^2) \tag{3.11}$$

$$G_\infty^{**}(B) = D + \rho V + j'_\infty \beta B' e^{\beta(1-j'_\infty)} \tag{3.12}$$

$$B' = \begin{cases} B + \sigma, & \text{exhaustive} \\ B, & \text{partially gated or gated} \end{cases}$$

$$\alpha = B + \sigma - D (\geq B - P > 0)$$

$$j_\infty = \max\{0, 1 - \frac{\rho V}{(1-\beta)\alpha}\}, \tag{3.13}$$

$$\psi_\infty^* = \frac{\beta V}{1-\beta} - (1 - j_\infty)\frac{\beta}{\rho}\alpha \tag{3.14}$$

$$j'_\infty = \begin{cases} 1, & \text{if } B'(1 - \frac{\beta}{2}) \leq g^*(B) \\ \frac{1-\sqrt{1-2\beta g^*(B)/B'}}{\beta} & \text{otherwise} \end{cases} \tag{3.15}$$

Outline of Proof. Let ψ_n^* and J_n equal ψ^* and J, respectively, from Lemma 3.1 when $N = n$. Using Lemma 3.1 it is straightforward to show that the the limits $j_\infty =$

$\lim_n \frac{J_n}{n}$ and $\psi_\infty = \lim_n \psi_n^*$ exist and (3.13) and (3.14) are true. Using Lemma 3.1 it is straightforward to show that $\lim_n \frac{G_n*(B)}{n}$ exists and (3.11) is true.

We now turn to (3.15). Let J_n' and $G_n^{**}(B)$ equal J' and $G^{**}(B)$, respectively, from Lemma 3.2 when $N = n$. Construct the function $Q_n(x)$ on the interval $[0, n]$ as follows. For all integer values x, let $Q_n(x) = \sum_{i=0}^{x-1}(\mu - \rho(i+1))\gamma$. Define the rest of the function $Q_n(x)$ by interpolating between the integer values using line segments. Let $Z_n = \max\{z \leq n : Q_n(Z_n) \leq G_n^*(B)\}$. It can be shown that $Q_n(x)$ is nondecreasing on $[0, n]$ (using the fact that $\mu - \rho(i+1) \geq 0$ if $i < N$), and this implies $|Z_n - J_n'| \leq 1$.

Let $q_n(x) = \frac{1}{n}Q_n(x \cdot n)$ and $q(x) = \lim_n q_n(x) = B'(x - x^2\beta/2)$. Let $z^* = \sup\{z \in (0,1) : q(z) \leq g^*(B)\}$. Note that $z^* = 1$ if $q(1) \leq g^*(B)$. Also, $z^* = (1/\beta)(1 - \sqrt{1 - 2\beta g^*(B)/B'})$ if $q(1) > g^*(B)$ because $q(0) = 0 < g^*(B)$, q is continuous, increasing, concave and quadratic. Using the fact that q and q_n are increasing one can show that $z^* = \lim_n Z_n/n$, which in turn equals $\lim_n J_n'/n$. Then (3.13) is implied.

We now turn to (3.12). Let Δ_k^n equal Δ_k in Lemma 3.2 when $N = n$. From Lemma 3.2, we can show that $\lim_n \sum_{i=0}^{n-1} \Delta_i^n = \frac{j_\infty' \beta B'}{\rho} e^{\beta(1-j_\infty')}$. This implies (3.12).

□

Theorem 3.1. Let G_∞^{**} be the function defined in Lemma 3.3. There is a unique value B_∞^* that satisfies $B_\infty^* \geq P$ and $G_\infty^{**}(B_\infty^*) = B_\infty^*$. The value B_∞^{**} is an upper bound for $\limsup_N P^*$.

Outline of Proof. Let $F(B) = B - G^{**}(B)$. Note that Problem LP is equivalent to finding the largest value B such that $F(B) \geq 0$. Let $F_n(B)$ and $G_n^{**}(B)$ equal $F(B)$ and $G_n^{**}(B)$ when $N = n$. Let $F_\infty(B) = \limsup_n F_n(B)$.

One can show that $G_n^{**}(\infty)$ is finite and at least $D + \rho V$ for all n, that $\limsup_n G_n^{**}(\infty)$ is also finite and at least $D + \rho V$. One can also show that $G_n^{**}(B)$ is also concave with B. Hence, F_n and F_∞ are convex functions that are finite, and $F_n(0) < 0 < F_n(\infty)$ for all n including $n = \infty$. Then there is a unique nonnegative value B_n^* such that $F_n(B_n^*) = 0$. Since F_n is increasing on $[0, \infty)$, $B_n^* \geq P^*$ for each finite n. Since F_∞ is increasing on $[0, \infty)$, $\lim_n B_n^* = B_\infty^*$. Then $B_\infty^* \geq \limsup_N P^* \geq P$.

From Lemma 3.3, $F_\infty(B) = B - G_\infty^{**}(B)$ if $B > P$. From the continuity of both $F_\infty(B)$ and $G_\infty^{**}(B)$, $F_\infty(B) = B - G_\infty^{**}(B)$ if $B = P$. The proof is completed by noting that B_∞^* is a unique solution that satisfies $F_\infty(B_\infty^*) = 0$ and $B_\infty^* \geq P$. □

4. Final Remarks

We will compare the bound given in Section 3 with the ones in [2] and [3]. Consider the case of $\beta \approx 1$ and $V << 1 - \beta$. If the service policy is exhaustive we also assume that $\sigma = 0$. Let $C = B_\infty^*/D$, where B_∞^* is the upper bound defined in Theorem 3.4. Then from the theorem we must have $C \approx 1 + C(1 - \sqrt{1 - 2/C})exp(\sqrt{1 - 2/C})$.

Hence, $C \approx 3.302$. This value of C is slightly lower than the ones in [1] and [2], which were 3.35 and 3.305, respectively.

Next we will discuss why the service policies are asymptotically optimal. We will use the lower bound $\limsup_N P^* \gtrsim \frac{\rho V}{1-\beta}$, which is true for all round-robin polling policies if ρ, V, β, and σ are constant with N. To see why this is a lower bound, consider large N and input traffic that are constants ρ. If W is the duration of a cycle of pollings then we expect on the average that $N\rho W = \mu(W - V)$, i.e., the amount of incoming traffic equals the amount of outgoing traffic during a cycle. Then we expect that there is a cycle such that $W \geq V/(1-\beta)$. There is a node corresponding to this cycle that has duration W between two successive pollings, so it collects $\rho V/(1-\beta)$ amount of data between the pollings.

Let $C = B^*_\infty[\rho V/(1-\beta)]^{-1}$, where B^*_∞ is defined in the Theorem 3.1. Using Theorem 3.1, Lemma 3.3, and the fact that $C \geq 1$, one can show that as $\beta \to 1$, $C \to 1$. Hence, the buffer requirements of the service policies are asymptotically optimal.

As a final remark, one can construct input traffic functions so that $\limsup_N P^* \geq \sigma[1 + e^{\beta Y}(e^\beta - 1 - \beta Y)] + \frac{\rho V}{1-\beta}$, where $Y = \max\{0, (e^\beta - 2)/\beta\}$.

Acknowledgements. The author would like to thank Moshe Sidi and Rene Cruz for stimulating discussions. The author would also like to thank Rene Cruz for pointing out errors in an earlier version.

References

[1] R. Cruz, "A calculus for network delay," *IEEE Trans. Info. Theory,* vol. 37, no. 1, pp. 114-141, Jan. 1991.

[2] I. Cidon, I. Gopal, G. Grover, and M. Sidi, "Real-time packet switching: A performance analysis," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1576-1586, Dec. 1988.

[3] G. Sasaki, "Input buffer requirements for round robin polling systems," *Proc. 27th Ann. Allerton Conf. Commun., Contr., and Computing,* Monticello, IL, pp. 397-406, Sept. 27-29, 1989.

[4] H. R. Gail, G. Grover, R. Guerin, S. L. Hantler, Z. Rosberg, M. Sidi, "Buffer size requirements under longest queue first," IBM Res. Rept. RC 16486 (#73187), Jan. 28, 1991.

[5] A. Birman, H. R. Gail, S. l. Hantler, Z. Rosberg, M. Sidi, "An optimal service policy for buffer systems," IBM Res. Rept. RC 16641(#73641), Mar. 5, 1991.

[6] P. R. Kumar and T. I. Seidman, "Distributed instabilities and stabilization methods in distributed real-time scheduling of manufacturing systems," *IEEE Trans. Auto. Contr.,* vol. 35, no. 3, pp. 289-298, Mar. 1990.

[7] C. H. Lu and P. R. Kumar,"Distributed scheduling based on due dates and buffer prioritization," Tech. Rept., Coordinated Science Laboratory, Univ. of Illinois at Urbana-Champaign, 1990.

[8] J. R. Perkins and P. R. Kumar,"Stable, distributed, real-time scheduling of flexible manufacturing/assembly/disassembly systems," *IEEE Trans. Auto. Contr.,* vol. 34, no. 2, pp. 139-148, Feb. 1989.

The Properties of Nearly-Periodic Tasks and Their Scheduling Model

Sungyoung Lee, Tzilla Elrad, and Deepak Dube

Department of Computer Science
Illinois Institute of Technology
Chicago, IL 60616
Email:CSELRAD@IITVAX.IIT.EDU

Abstract

We introduce a new type of task, namely, "nearly-periodic task", which is a superset of periodic tasks, and informally covers a class of task whose arrival behavior is repetitive, but its interarrival time varies within certain boundary. The objective of this new class of task is motivated by the fact that classical execution tasks for hard real-time systems are not sufficient for describing certain computational requirements. We also consider the scheduling model for nearly-periodic task using a visualization language, Statecharts, to determine timing error in systems.

1. Introduction

Real-time systems are widely used to control, monitor or perform critical operations, and to respond quickly to emergency events in a wide range of embedded applications. Such systems are required to process tasks with stringent timing requirements and must perform these tasks in such a way that these timing requirements are guaranteed to be met [4].

In areas of real-time scheduling theory, the common approach is to define the execution model as a set of preemptive and/or nonpreemptive periodic and aperiodic tasks. In real-time systems, however, these conventional tasks are not enough to explain characteristics of a task whose arrival behavior is repetitive but interarrval time varies within a limited boundary. Thus, we introduce a new type of task, namely, nearly-periodic task to cope with the addressing requirements. A nearly-periodic task can be clearly distinguished from an aperiodic task whose arrival behavior is totally random and a sporadic task [5] whose arrival behavior is same as an aperiodic task except that it has a minimum interarrival time. We classify a nearly-periodic task as a strict nearly-periodic task and a non strict-nearly periodic task, and address their scheduling considerations. The former case, we can consider its scheduling within a rate monotonic scheduling [2] framework which is of great practical importance in real-time systems [3] while the latter case we can apply aperiodic tasks scheduling algorithms. We also model the scheduling behavior of nearly-periodic tasks using a visualization language, Statecharts, to determine timing error in systems.

The rest of this paper is organized as follow. Section 2 defines a nearly-periodic task and compares it with conventional real-time tasks and show its relationships with other tasks. Section 3 define a strict nearly-periodic task and a non-strict nearly-periodic task, and describe their scheduling considerations. Section 4 consider the scheduling model of nearly-periodic tasks using Statecharts. We conclude the paper in section 5.

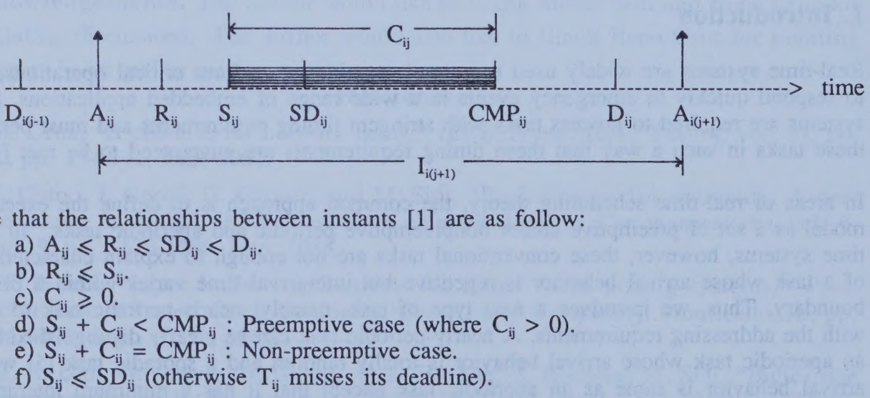
2. Real-Time Tasks

2.1 Terminology and Notation

We have used the terminology and notation from the Muntz's works [1]. The timing constraints of task i are a set of n_i predicates, i.e., $T_i = \{T_{ik} \mid k = 1..n_i\}$ where $n_i > 0$. Each T_{ik} is a function of an ordered tuple of time instants $\{A_{ij}, R_{ij}, S_{ij}, SD_{ij}, CMP_{ij}, D_{ij}\}$ and an ordered tuple of time intervals $\{C_{ij}, I_{ij}\}$ where:

1) T_{ij} = j^{th} instance of task i.
2) A_{ij} = arrival time of T_{ij}.
3) R_{ij} = ready time of T_{ij}.
4) S_{ij} = start time of execution of T_{ij} (j^{th} instance of task i cannot start before R_{ij}).
5) SD_{ij} = scheduling deadline (j^{th} instance of task i must start before this time otherwise it cannot meet deadline. T_{ij} must be scheduled for execution in order to meet its deadline).
6) CMP_{ij} = completion time of execution of T_{ij}.
7) D_{ij} = deadline of T_{ij}.
8) C_{ij} = execution time of T_{ij}.
9) I_{ij} = interarrival time between two instances of task i where $j \geqslant 1$ ($I_{ij} = A_{ij} - A_{i(j-1)}$).

Figure 2-1) A task instance of non-preemptive



Note that the relationships between instants [1] are as follow:

a) $A_{ij} \leqslant R_{ij} \leqslant SD_{ij} \leqslant D_{ij}$.
b) $R_{ij} \leqslant S_{ij}$.
c) $C_{ij} \geqslant 0$.
d) $S_{ij} + C_{ij} < CMP_{ij}$: Preemptive case (where $C_{ij} > 0$).
e) $S_{ij} + C_{ij} = CMP_{ij}$: Non-preemptive case.
f) $S_{ij} \leqslant SD_{ij}$ (otherwise T_{ij} misses its deadline).

2.2. Definition of Scheduling Task Attribute.

We define the notion of Periodic, Aperiodic, Sporadic, and Nearly-periodic tasks using the notations defined in the section 2.1.

1) Periodic task: task i is a periodic task if

a) $I_{ij} = k_{periodic}$ for every j where $k_{periodic}$ is constant and $k_{periodic} > 0$.
b) $A_{ij} = R_{ij}$.

2) Aperiodic task: task i is an aperiodic task if

a) $I_{ij} \neq k_{aperiodic}$ for every j where $k_{aperiodic}$ is any constant and $k_{aperiodic} > 0$ (It is possible to have a zero interarrival time, thus $I_{ij} = C_{ij} = k_{aperiodic} = 0$).

3) Sporadic task: task i is a sporadic task if

a) $A_{ij} = R_{ij}$.
b) $I_{ij} \geqslant k_{sporadic}$ for every j where $k_{sporadic}$ is the minimum interarrival time and $k_{sporadic} \geqslant C_{ij}$.

4) Nearly-periodic task: task i is a nearly-periodic task if

$k_{nearly\text{-}periodic} - e_{lower\text{-}bound} \leqslant I_{ij} \leqslant k_{nearly\text{-}periodic} + e_{upper\text{-}bound}$
where (1) $k_{nearly\text{-}periodic}$ is the expected value of interarrival time of task i (= I_i).
Thus, if we define I_i to be the interarrival time of task i, then:

$$k_{nearly\text{-}periodic} = E[I_i] = (\sum_{j=1}^{m} I_{ij})/m \text{ for a large m, } E[I_i] \text{ is expected value of } I_i.$$

(2) $0 \leqslant e_{lower\text{-}bound} \leqslant k_{nearly\text{-}periodic}$, $0 \leqslant e_{upper\text{-}bound}$.
(3) $VAR[I_{ij}] = e_{lower\text{-}bound} + e_{upper\text{-}bound}$ where $VAR[I_{ij}]$ is variance of I_{ij}.

2.3. Discussion

In order to clearly distinguish the behavior of nearly-periodic task from the other tasks, we consider the following lemmas to show relationships.

Lemma 1: The nearly-periodic task i is a periodic task if

a) $R_{ij} = A_{ij}$
b) $e_{lower\text{-}bound} = e_{upper\text{-}bound} = 0$.

Proof:
It follows trivially from the definition:
$e_{lower\text{-}bound} = e_{upper\text{-}bound} = 0$, $k_{nearly\text{-}periodic} \leqslant I_{ij} \leqslant k_{nearly\text{-}periodic}$ implies $I_{ij} = k_{nearly\text{-}periodic}$ --- (1.1)
Equation (1.1) and lemma 1-a) imply that task i is a periodic task.

Lemma 2: A nearly periodic task i is a sporadic task if

a) $R_{ij} = A_{ij}$
b) $k_{sporadic} = k_{nearly\text{-}periodic} - e_{lower\text{-}bound}$
c) $e_{upper\text{-}bound}$ -> infinite.

Proof:
By definition of nearly-periodic task $I_{ij} \geqslant k_{nearly\text{-}periodic} - e_{lower\text{-}bound}$ --- (2.1).
From lemma 2-b) and equation (2.1), we get $I_{ij} \geqslant k_{sporadic}$ --- (2.2).
From (2.2) and lemma 2-a), the nearly-periodic task satisfies the conditions of a sporadic task.

Lemma 3: A nearly-periodic task i is aperiodic task if

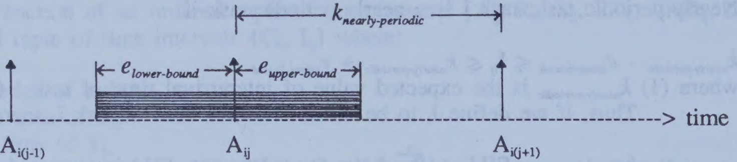
a) $k_{nearly\text{-}periodic} = e_{lower\text{-}bound}$.
b) $e_{upper\text{-}bound}$ -> infinite.

Proof:
Condition a) and b) of lemma 3 implies $0 \leqslant I_{ij} \leqslant$ infinite --- (3.1)
Nearly-periodic task does not have constant interarrival time by the definition --- (3.2)
From (3.1) and (3.2) the lemma follows.

3. Nearly-Periodic Tasks

3.1. Property of a Nearly-Periodic Task

The time event diagram below shows arrival behavior of a nearly-periodic task i.



The shadow area represents the variation in interarrival time of each instance of nearly-periodic task i. A nearly-periodic task is like a periodic task except that its constant period may be varied based on the workload. A nearly-periodic task tends to behave as a periodic task if $VAR[I_i] \rightarrow 0$. Also, a nearly-periodic task has following characteristics;

a) A scheduler can predict the next arrival unlike aperiodic tasks case (In a strong sense, we can expect A_{ij+1} within a certain bound more accurately after A_{ij}. In periodic tasks case, however, any A_{ij} are known a priori).

b) R_{ij} may not be equal to A_{ij} and C_{ij} may vary.

c) The distribution of task's interarrival time is (truncated) normal satisfying $k_{nearly\text{-}periodic} - e_{lower\text{-}bound} \leqslant I_{ij} \leqslant k_{nearly\text{-}periodic} + e_{upper\text{-}bound}$.

3.2. Strict Nearly-Periodic Tasks

When we allocate a set of nearly-periodic tasks to a given set of periodic tasks to schedule, they can be classified into two categories; one contains strict nearly-periodic tasks, the other contains non-strict nearly-periodic tasks. In this section, we define strict nearly-periodic tasks and describe a scheduling consideration for such tasks.

3.2.1. Definition

We define strict nearly-periodic task in an environment of a periodic task set. Consider a pool of n periodic tasks, scheduled as per rate monotonic algorithm [2] with priority P_i where $1 \leqslant i \leqslant n$. Task T_{np}, with interarrival time I_{np}, qualifies to be a nearly-periodic task if the following algorithm yields set = true:

```
$P_i = i \ \forall \ i \in \{ 1,..., n\}$
/* We assume that tasks are arranged in ascending order of their interarrival times;
   lower the value of $P_i$, the higher the priority */
set = true; /* initialization */
found = false; /* initialization */
DO i = 1, n  /* for all periodic tasks */
   IF (NOT found) AND $E[I_{np}] < I_i$ THEN
      { found = true;
        $P_{np}$ <- i } /* assign the right place to $P_{np}$ */
   IF found THEN $P_i$ <- i + 1;
ENDDO
IF (($E[I_{np}]$ - $VAR[I_{np}]$) < $I_{(i-1)}$) AND ($E[I_{np}]$ < $VAR[I_{np}]$) THEN set = false
/* compare $E[I_{np}]$ with $I_{(i-1)}$ */
IF (($E[I_{np}]$ + $VAR[I_{np}]$) > $I_{(i+1)}$) AND ($E[I_{np}]$ < $VAR[I_{np}]$) THEN set = false
/* compare $E[I_{np}]$ with $I_{(i+1)}$ */
```

Note that the execution priority of a strict nearly-periodic task does not change because its interarrival time is bounded.

3.2.2. Scheduling Consideration for Strict Nearly-Periodic Tasks

Rate monotonic scheduling algorithms have many implementation advantages in hard real-time system environments [3]. Since the rate monotonic scheduling algorithm requires that periodic tasks be given highest priority, a periodic task i is ready for execution immediately upon arrival. This is frequently included in the definition of periodic tasks in the literature. The major problem of nearly-periodic task scheduling is, however, that the arrival time of a nearly-periodic task may be different from its ready time. Suppose, given a set of periodic tasks that is schedulable based on the rate monotonic algorithm. The algorithm requires that the ready time coincide with the arrival time. However, the nearly-periodic task's arrival behavior violates this requirement since R_{ij} may not be A_{ij}. Thus we must consider a scheduling scheme for a strict nearly-periodic task to apply a rate monotonic scheduling algorithm as follows;

a) If the arrival time of j^th instance of a strict nearly-periodic task i is same as its ready time and its execution time is known then simply apply the rate monotonic algorithm.

b) Upon arrival of a strict nearly-periodic task i, it is assigned a priority as per the algorithm discussed in section 3.2.1. Two scenarios are now possible depending on the ready time of the strict nearly periodic task:

i) The j^th instance of strict nearly periodic task i is not ready for execution ($A_{ij} \neq R_{ij}$). In this case, lower priority periodic tasks may continue execution till R_{ij}. At this time, these low priority periodic tasks are preempted and execution for strict nearly periodic task T_{ij} starts ($S_{ij} = R_{ij}$). If, however, at R_{ij}, a high priority periodic task is undergoing execution, then T_{ij} waits till all the higher priority periodic tasks have finished execution ($S_{ij} > R_{ij}$).

ii) Suppose the higher priority periodic tasks finish execution time Δ after A_{ij}. If A_{ij} Δ + < R_{ij}, then the scheduler would schedule other lower priority periodic tasks till R_{ij}.

3.3. Non-Strict Nearly-Periodic Task

3.3.1 Background

In this section, we define a non-strict nearly-periodic task based on the statistical behavior of its requests and describe scheduling consideration for such tasks. The following figures represent the distributions of I_i for the periodic tasks, aperiodic tasks, sporadic tasks, strict nearly-periodic tasks, and non-strict nearly-periodic tasks.

Figure 3-1) Periodic tasks

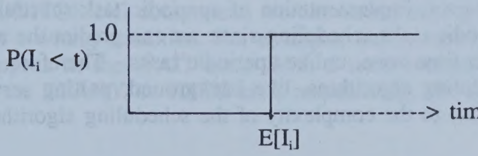


Figure 3-2) Aperiodic tasks

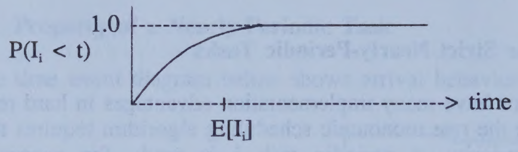


Figure 3-3) Sporadic tasks

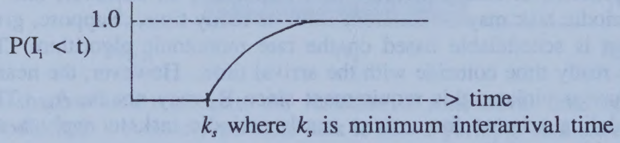


Figure 3-4) Strict nearly-periodic tasks

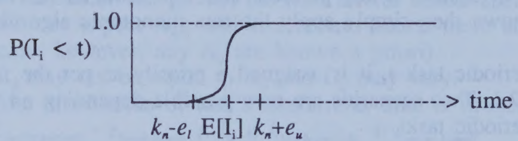
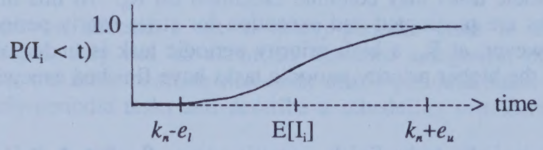


Figure 3-5) Non-strict nearly-periodic tasks



3.3.2 Definition

A nearly-periodic task is called a non-strict nearly-periodic task if $VAR[I_i]=E[I_i]/(1 + \varsigma)$ where ς is a small positive quantity. If ς is large, then the task becomes strictly nearly-periodic task.

3.3.3 Scheduling Considerations

Even though the property of non-strict nearly-periodic tasks is repetitive, if the value of is a very small positive quantity, then the arrival behavior of non-strict nearly-periodic tasks is similar to aperiodic tasks. Thus we must consider applying aperiodic task scheduling algorithms to the non-strict nearly-periodic tasks rather than applying the rate monotonic scheduling. In this case, however, implementation of aperiodic task scheduling is simpler than the general case of aperiodic tasks scheduling since we can predict the next arrival of nearly-periodic tasks in certain time zone, unlike aperiodic tasks. Therefore, we can choose simple aperiodic task scheduling algorithms, like background, polling server, or deferrable server, to avoid time lost due to the complexity of the scheduling algorithms.

4. Scheduling Model of Nearly-Periodic Tasks

We have discussed the properties of nearly-periodic tasks and their scheduling considerations. In this section, we turn our focus to the scheduling model of nearly-periodic tasks using a visualization language, Statecharts. There are many modeling tools available. However, we chose the Statecharts [7] since it has precise semantic, integrated view, and a function for timing specifications.

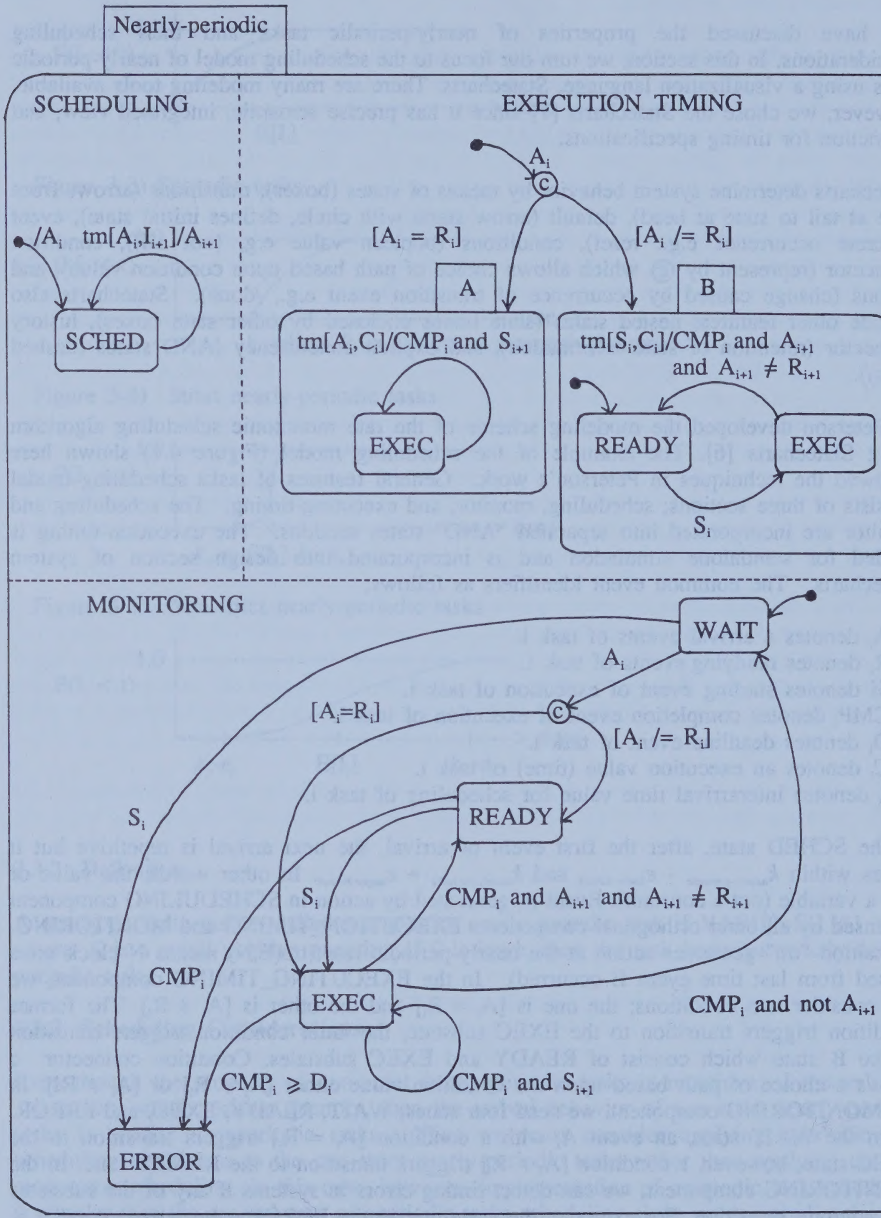
Statecharts determine system behavior by means of states (boxes), transitions (arrows from state at tail to state at head), default (arrow starts with circle, defines initial state), event (discrete occurrence e.g., reset), conditions (boolean value e.g, [not all]), condition connector (represent by © which allows choice of path based upon condition value), and actions (change caused by occurrence of transition event e.g., /done). Statecharts also include other features; nested states (state boxes enclosed by other state boxes), history connector (retention of state information), and explicit concurrency (AND states (dashed lines)).

A. Peterson developed the modeling scheme of the rate monotonic scheduling algorithm using Statecharts [6]. The example of the scheduling model (Figure 4.1) shown here followed the techniques in Peterson's work. General features of tasks scheduling model consists of three sections; scheduling, monitor, and executing-timing. The scheduling and monitor are incorporated into separated "AND" states sections. The execution-timing is needed for standalone simulation and is incorporated into design section of system Statecharts. The common event identifiers as follows;

1) A_i denotes a arrival events of task i.
2) R_i denotes readying events of task i.
3) Si denotes starting event of execution of task i.
4) CMP_i denotes completion event of execution of task i.
5) D_i denotes deadline event of task i.
6) C_i denotes an execution value (time) of task i.
7) I_i denotes interarrival time value for scheduling of task i.

In the SCHED state, after the first event of arrival, the next arrival is repetitive but it varies within $k_{nearly\text{-}periodic} - e_{lower\text{-}bound}$ and $k_{nearly\text{-}periodic} + e_{upper\text{-}bound}$. In other words, the value of I_i is a variable (not a constant). Event A_i generated by action in SCHEDULING component is sensed by all other orthogonal components EXECUTION_TIMING and MONITORING. Transition "tm" generates action at the nearly-periodic rate (tm(E,N) means N clock units passed from last time event E occurred). In the EXECUTING_TIMING component, we can consider two conditions; the one is $[A_i = R_i]$ and the other is $[A_i \neq R_i]$. The former condition triggers transition to the EXEC substate, the latter condition triggers transition to the B state which consist of READY and EXEC substates. Condition connector c allows a choice of path based upon the condition value either $[A_i = R_i]$ or $[A_i \neq Ri]$. In the MONITORING component, we need four states, WAIT, READY, EXEC, and ERROR. From the WAIT state, an event A_i with a condition $[A_i = R_i]$ triggers transition to the EXEC state, however, a condition $[A_i \neq R_i]$ triggers transition to the READY state. In the MONITORING component, we can detect timing errors in systems if any of the substates has a wrong transition. For example, the system is in the ERROR state when an event S_i happens in the WAIT state. With a same reason, if an event CMP_i or $CMP_i \geqslant D$ occurs in the READY or EXEC state respectively, the system suffers timeing error.

Figure 4-1) Timeline and characteristics


Nearly-periodic
SCHEDULING
EXECUTION_TIMING
A_i
C
/A_i
tm[A_i,I_{i+1}]/A_{i+1}
[A_i = R_i]
[A_i /= R_i]
A
B
SCHED
tm[A_i,C_i]/CMP_i and S_{i+1}
tm[S_i,C_i]/CMP_i and A_{i+1} and A_{i+1} ≠ R_{i+1}
EXEC
READY
EXEC
S_i
MONITORING
WAIT
A_i
[A_i=R_i]
C
[A_i /= R_i]
S_i
READY
S_i
CMP_i and A_{i+1} and A_{i+1} ≠ R_{i+1}
CMP_i
EXEC
CMP_i and not A_{i+1}
CMP_i ≥ D_i
CMP_i and S_{i+1}
ERROR

5. Summary

We defined a new type of task, nearly-periodic task, in real-time systems and considered the relationships with other conventional real-time tasks, and addressed scheduling considerations within the rate monotonic scheduling framework. Also, we modeled scheduling of nearly-periodic tasks using Statecharts to determine timing error in systems.

REFERENCES

[1] Muntz, A.H. and Horowitz, E., "A Framework for specification and Design of Software for Advanced Sensor Systems", IEEE Real-Time Systems Symposium, 1989, pp 204-213.

[2] Liu, C.L. and Layland J.W., "Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment", Journal of ACM 20 (1) No.1 Jan. 1973, pp 46-61.

[3] Sha, L., Lehoczky, J. P. and Rajkumar, R. "Solutions for Some Practical Problems in Prioritized Preemptive Scheduling," IEEE Real-Time Systems Symposium, 1986, pp 181-191.

[4] Rajkumar, R., Task Synchronization in Real-Time Systems, Ph.D. Dissertation, Department of Electrical and Computer Engineering, Carnegie Mellon University, August 1989.

[5] Mok, A.K., Fundamental Design Problems for the Hard Real-Time Environment, MIT Ph. D. Dissertation, Cambridge, MA. May, 1983.

[6] Peterson, S., "Reusable Requirements using Statecharts," Software Engineering Institute, Carnegie Mellon University, October, 1989.

[7] Harel, D., et al. "STATEMATE: A Working Environment for the Development of Complex Reactive Systems," IEEE Transaction on Software Engineering, Vol. 16. No. 4. April 1990. pp 403-414.

Analysis of a Fair, Slotted Ring with Spatial Reuse

Ilan Kessler & Arvind Krishna
IBM Thomas J. Watson Research Center
Yorktown Heights, NY 10598

Abstract

In this work a simple fairness mechanism for slotted rings with spatial reuse is investigated. The fairness mechanism is related to MetaRing of IBM and Orwell of British Telecom. We define a mathematical model for a slotted ring with spatial reuse and a fairness mechanism, and obtain an exact expression for the average time taken to serve each station exactly once. This expression can be evaluated in $O(N^2)$ time for a ring with N nodes. It is shown that this expression is equal to $N/2 + o(N)$, where the slot takes unit time to complete one circulation of the ring. Since $N/2$ is a lower bound on the time taken to serve all N nodes, this shows that the cost of fairness in terms of throughput is small. The analysis depends on combinatorial arguments and a ballot theorem.

1 Introduction

Many current LAN designs, such as Token Ring, Cambridge Ring, and FDDI [1, 2], are based on ring topologies. In order to achieve simplicity and fairness, these designs avoid concurrent access with spatial reuse of the LAN. Spatial reuse is the concurrent usage of distinct segments of the ring. Lately, there has been a resurgence of interest in the use of spatial bandwidth reuse [3, 4, 5, 6]. The advantage of spatial reuse lies in the increase of network capacity for the same infrastructure. In fact, when traffic is homogenous a factor of two is gained in the throughput of a unidirectional ring, since packets have to traverse only half the ring on average. Unfortunately, the introduction of spatial reuse may cause nodes to suffer long delays in accessing the ring. This problem is referred to as *starvation.* In order to avoid this problem, *fairness* mechanisms have been introduced [4, 5, 6].

An important performance issue is understanding the impact of fairness on network throughput. No analytical results regarding this issue are known to the authors. The performance of slotted rings with spatial resue and a fairness mechanism has been considered in [7, 8]. The analysis in [8] approximates a fair, slotted ring with spatial reuse as a polling system, and uses techniques from polling systems to find performance results. The validity of this approximation is not clear, as the fairness mechanism is not taken into account in the polling systems model. More work in a similar spirit has been done in [7]. The approximation there also ignore the fairness mechanism for analytical purposes, but the author mentions that exact results are possible for the special case in which there are at most two active nodes on the ring. Ignoring the fairness mechanism for performance analysis is justifiable as a light traffic approximation, but not for the case of heavy traffic.

Our goal is to find the maximum throughput of a fair, slotted ring with spatial reuse. To this end, we model a simple fairness mechanism which is almost identical to the Orwell protocol [5]. In brief, the fairness mechanism ensures that no node gets a second chance to transmit a packet until every node has had at least one chance. The model is described in detail in Section 2. In Section 3, we begin the analysis of this model. First, an exact expression for the time taken to serve each station exactly once is derived. This expression can be used to compute the network throughput, and we compare the throughput of our model with that of a Token Ring network. As mentioned above, without imposing any fairness constraints, spatial reuse gains a factor of two in throughput. In Section 4, we show that the penalty of our simple fairness mechanism disappears asymptotically with the number of nodes in the ring.

2 System and Model

In this section we first describe the network under consideration and the fairness mechanism it employs. Thereafter we present the mathematical model used for the analysis.

2.1 System

Consider a communication network operating as follows. The nodes in the network are arranged in a ring. Each node generates packets to be transmitted to other nodes in the network. All packets are of equal length. There is one slot circulating around the ring, which can accommodate one packet at a time. The nodes transmit by writing their packets into the slot, and receive by reading the content of the slot. The slot can be either occupied (i.e. contain a packet) or free (i.e. contain no packet). The slot consists of three parts – the status field, the address field and the data field. The status field indicates whether the slot is occupied or free. Each node in the network can modify this field, thus marking the slot as occupied or free. The address field contains the destination address of the packet carried by the slot. The data field contains the packet carried by the slot. Both the address and the data fields are meaningful only if the slot is marked as occupied.

In order to transmit a packet, a node must have a *quota*. A separate quota is required for the transmission of each packet. We assume that the nodes always have packets to transmit. The operation of the network is carried out in *cycles*. At the beginning of each cycle, each node gets one quota. During the cycle, the nodes transmit their packets according to the rules described below. The cycle ends when all N nodes have used their quotas (i.e. each node has transmitted one packet). During a cycle, no new quotas are granted to the nodes.

The mechanism employed by the nodes during a cycle is as follows. Whenever the slot arrives to a node, the node examines the status field of the slot and carries out the following two steps:

1. If the slot is occupied and the destination of the packet on the slot is the current node, then the node copies the packet into its own buffer, and marks the slot as free.
2. If the slot is free and the node has yet to transmit its own packet, then the node marks the slot as occupied and copies its own packet and the destination address of the packet into the data and address fields of the slot, respectively.

Note that step (2) is carried out after step (1). This means that if the slot arrives occupied to a destination node that has yet a packet to transmit, upon departing from this node the slot is occupied by the packet of this node. A packet destined to its source node has to travel all the way around the ring (a broadcast message).

2.2 Model

The network described above is modeled as follows. Consider a ring with N nodes equally spaced on it. Initially, each node has one packet to be transmitted to one of the N nodes. The destination of each packet is chosen equiprobably amongst all N nodes, independently of all other packets. A single slot that can carry exactly one packet circulates around the ring with a constant speed. The slot can be either free or occupied. If the slot arrives free to a node that has a packet to transmit, it picks up this packet and becomes occupied. The slot becomes free again only upon arriving to the destination of the packet it carries (where it can pick up a new packet if this node has yet one to transmit). A packet that has been carried by the slot all the way from the source node to the destination node is said to be *delivered.* A node whose packet has been picked up by the slot is said to be *empty* (even if the packet has not been delivered yet). Accordingly, a node that has yet a packet to transmit is said to be *full.* Empty nodes do not become full again. The time it takes the slot to complete one circulation of the ring is taken to be the unit time.

The slot starts at a node that is chosen equiprobably amongst all nodes. This is equivalent to putting a *dummy* packet on the slot whose destination is chosen equiprobably amongst all N nodes, and letting the network start operating with the delivery of this dummy packet. The dummy packet is referred to as packet 0. The next packet to be carried by the slot, which is the packet starting at the node that is the destination of the dummy packet, is packet 1. In general, packet k is the packet carried by the slot after it has carried packet $k-1$. The *evacuation time*, T_{evac}, of the network is defined to be the time taken to deliver packets $1, \ldots, N$ to their destinations. Note that since the slot moves with a (constant) unit speed, time and distance are interchangable.

For convenience, we number the nodes as $1, 2, \ldots, N$ in the direction of slot movement, arbitrarily picking some node as node 1. The slot moves from a node i to node $i \oplus 1$, where $i \oplus 1 = i + 1$ if $1 \leq i < N$, and $i \oplus 1 = 1$ if $i = N$. In general, if the slot is at node i and moves forward by k nodes it is at node $i \oplus k$, where $i \oplus k$ is defined inductively on k by $i \oplus k = (i \oplus (k-1)) \oplus 1$. In a similar manner, define $i \ominus k = i \oplus (N - k)$ for

$0 \leq k \leq N$. Note that since the slot circulates at unit speed, it takes k/N time to go from node i to node $i \oplus k$ for $1 \leq k \leq N$.

3 Evacuation Time

In this section we derive an expression for the expected evacuation time of the ring. The evacuation time has a direct relation to the throughput of the ring, as discussed at the end of this section.

Let d_m be the distance between the source and destination of packet m, and let f_m be the distance between the destination of packet m and the source of packet $m+1$. It follows from the definition of d_m and f_m that T_{evac} is given by

$$T_{evac} = \sum_{m=1}^{N} (d_m + f_m), \tag{1}$$

where by definition, $f_N = 0$. The expected value of d_m is $(1 + 1/N)/2$, since it is equiprobable that d_m takes any of the N values $1/N, 2/N, \ldots, 1$. Hence, from Equation (1)

$$\mathrm{E}[T_{evac}] = \sum_{m=1}^{N} \mathrm{E}[d_m + f_m] = \frac{N+1}{2} + \sum_{m=1}^{N-1} \mathrm{E}[f_m].$$

Thus, determining $\mathrm{E}[T_{evac}]$ reduces to determining $\mathrm{E}[f_m]$. Most of this section is devoted to finding an exact expression for $\mathrm{E}[f_m]$.

After packets $1, 2, \ldots, m$ have been picked up, there are at most m contiguous empty nodes. Therefore, $\mathrm{E}[f_m] = \left[\sum_{k=1}^{m} k \Pr(f_m = k/N)\right]/N$. Conditioning on the destination of packet m,

$$\Pr(f_m = \frac{k}{N}) = \sum_{i=1}^{N} \Pr\begin{pmatrix}\text{Destination of packet}\\ m \text{ is node } i\end{pmatrix} \Pr\left(f_m = \frac{k}{N} \middle| \begin{array}{l}\text{Destination of packet}\\ m \text{ is node } i\end{array}\right).$$

Definition: There is a *chain* of length l starting at node i if the l contiguous nodes $i, i \oplus 1, \ldots i \oplus (l-1)$ are empty and the nodes on either side of these contiguous nodes, namely nodes $i \ominus 1$ and $i \oplus l$, are full.

This definition implies that

$$\Pr\left(f_m = \frac{k}{N} \middle| \begin{array}{l}\text{Destination of packet}\\ m \text{ is node } i\end{array}\right) = \Pr\left(\bigcup_{l=k}^{m} \left\{\begin{array}{l}\text{A chain of length } l \text{ starts}\\ \text{at node } i \ominus (l-k) \text{ after}\\ \text{packets } 1, 2, \ldots, m \text{ have}\\ \text{been picked up by the slot}\end{array}\right\}\right), \tag{2}$$

because of the following two facts. The first fact is that the only way $f_m = k/N$ given that packet m was destined to node i is if there is a chain that contains node i and stops at node $i \oplus (k-1)$. The second fact is that the destination of packet m is independent of everything in the past of the system, hence its destination i is independent of the event that i is part of a particular chain. For fixed m, k, and i, let $\mathcal{A}_l$ be the event that there is a chain of length l starting at node $i \ominus (l-k)$ after packets $1, 2, \ldots, m$

have been picked up by the slot. The occurrence of $\mathcal{A}_l$ excludes the occurrence of $\mathcal{A}_j$ if $j \neq l$ because the occurrence of $\mathcal{A}_l$ implies that node $i \ominus (l-k+1)$ is full and nodes $i \ominus (l-k), i \ominus (l-k-1), \ldots, i$ are empty. Thus, the events $\mathcal{A}_l$ for $k \leq l \leq m$ are mutually disjoint and Equation (2) simplifies to

$$\Pr\left(f_m = \frac{k}{N} \middle| \begin{array}{l}\text{Destination of packet} \\ m \text{ is node i}\end{array}\right) = \sum_{l=k}^{m} \Pr\left(\mathcal{A}_l\right). \tag{3}$$

Before proceeding with the evaluation of $\Pr(\mathcal{A}_l)$, we give a physical description of how $\mathcal{A}_l$ occurs. In order to have a chain of length l starting at node $a = i \ominus (l-k)$ the following must occur: (i) l packets are delivered to nodes $a, a \oplus 1, \ldots, a \oplus (l)$ such that all of them get empty except for node $a \oplus l$, and (ii) $m - l$ packets are delivered to nodes $a \oplus (l+1), a \oplus (l+2), \ldots, a \ominus 1$ such that node N is full. A node is empty if and only if it has been the source of some packet. The source of packet m depends only on the destination of packet $m-1$ and the sources of packets $1, 2, \ldots, m-1$. It follows that the source of packet m can be determined from the destinations *only* of packets $0, 1, \ldots, m-1$. We now proceed with formalizing this description.

We define a collection of random variables $X(m,i)$ as follows. Let $X(m,i)$ be the number of packets that were destined to node i out of packets $0, 1, \ldots, m-1$. Since the destination of packet $k+1$ is independent of the destinations of packets $1, 2, \ldots, k$ it is evident that $X(m,i), 1 \leq i \leq N$, are interchangeable random variables. The following lemma, stated without proof[1], shows how $X(m,i)$ can be used to determine whether a particular node is empty or full.

Lemma 1 *Node i is full after packets* $1, 2, \ldots m$ *have been picked up by the slot if and only if* $\sum_{j=0}^{r-1} X(m, i \ominus j) < r$ *for all* r, $1 \leq r \leq N$.

Lemma 1 implies that $\mathcal{A}_l$ occurs if and only if the following three conditions are met:

(a) $\displaystyle\sum_{j=0}^{r-1} X(m, a \oplus l \ominus j) < r$ for $1 \leq r \leq l+1$,

(b) $\displaystyle\sum_{j=0}^{r-1} X(m, a \ominus 1 \ominus j) < r$ for $1 \leq r \leq N-(l+1)$, and

(c) $\displaystyle\sum_{j=0}^{l} X(m, a \oplus j) = l$,

where $a = i \ominus (l-k)$. Conditions (a) and (b) together are necessary and sufficient for nodes $a \ominus 1$ and $a \oplus l$ to be full. If conditions (a) and (b) are satisfied, then condition (c) is necessary and sufficient for nodes $a, a \oplus 1, \ldots, a \oplus (l-1)$ to be empty. Thus,

$$\Pr\left(\mathcal{A}_l\right) = \Pr\left(\text{(c) is true}\right) \Pr\left(\text{(a) is true} \mid \text{(c) is true}\right) \Pr\left(\text{(b) is true} \mid \text{(a) and (c) are true}\right). \tag{4}$$

The second and third probabilities in the equation above can be evaluated by an application of the *Ballot Theorem*[2] [11]. For convenience, the Ballot Theorem is stated below.

---

[1] Similar results can be found in [9, Equation 3] and [10, pp. 406].

[2] The Ballot Theorem was used in a similar manner in [9].

Theorem 1 *Let $x_1, x_2, \ldots, x_n$ be interchangeable random variables taking non-negative integer values. Then,*

$$\Pr\left(x_1 + x_2 + \cdots + x_r < r \text{ for } r = 1, 2, \ldots, n \,\middle|\, \sum_{i=1}^{n} x_i = \alpha \leq n\right) = 1 - \frac{\alpha}{n}.$$

Since the destinations of packets are independent of each other, it follows that $\Pr((\mathrm{c}) \text{ is true}) = N^{-m}\binom{m}{l}(l+1)^l(N-l-1)^{m-l}$. The second probability on the right hand side of Equation (4) can be computed by a direct application of Theorem 1 to be $(l+1)^{-1}$. The third and last probability on the right side of Equation (4) can also be computed from Theorem 1 after making the following observation. Conditioning on the value of $\sum_{i=a\oplus(l+1)}^{a\ominus 1} X(m,i)$, the random variables $X(m, a \oplus (l+1)), X(m, a \oplus (l+2)), \ldots, X(m, a \oplus (N-1))$ are independent of $X(m,a), X(m, a \oplus 1), \ldots, X(m, a \oplus l)$. Condition (c) is equivalent to the condition that $\sum_{i=a\oplus(l+1)}^{a\ominus 1} X(m,i) = m - l$. Hence, the last probability is equal to $\Pr((\mathrm{b}) \text{ is true} \mid \sum_{i=a\oplus(l+1)}^{a\ominus 1} X(m,i) = m - l)$, and this can be directly evaluated from Theorem 1. These computations yield

$$\Pr(\mathcal{A}_l) = N^{-m}\binom{m}{l}(l+1)^l(N-l-1)^{m-l}(l+1)^{-1}\left(1 - \frac{m-l}{N-l-1}\right).$$

Simplifying the above expression and substituting into Equation (3) yields

$$\Pr\left(f_m = \frac{k}{N}\right) = \sum_{l=k}^{m} N^{-m}\binom{m}{l}(l+1)^{l-1}(N-l-1)^{m-l}\left(1 - \frac{m-l}{N-l-1}\right). \tag{5}$$

Equation (5) implies that

$$E\left[T_{evac}\right] = \frac{N+1}{2} + \sum_{m=1}^{N-1}\sum_{k=1}^{m}\frac{k}{N}\sum_{l=k}^{m} N^{-m}\binom{m}{l}(l+1)^{l-1}(N-l-1)^{m-l}\left(1 - \frac{m-l}{N-l-1}\right). \tag{6}$$

Equation (6) can easily be used to numerically compute the average value of T_{evac}. A straightforward computation would take $O(N^3)$ steps, but a little thought shows that by exchanging the summation on k and l the amount of computation can be reduced to $O(N^2)$ steps.

The throughput of the ring is defined to be N divided by the expected time it takes to deliver N packets, $E[T_{evac}]$. We compare the throughput of our model with that of a token-ring network. As in our model, assume that the propagation delay around the ring is one time unit. Then, each node holds the token for at least unit time, and then passes the token to the next node. Thus, the total time to serve all N nodes is $N+1$, which implies that the throughput is $1 - 1/(N+1)$. Figure 1 plots the ratio of the throughput of our model and the throughput of a token-ring against the number of nodes on the ring. The graph shows that the ratio of the throughputs is tending to two, which implies

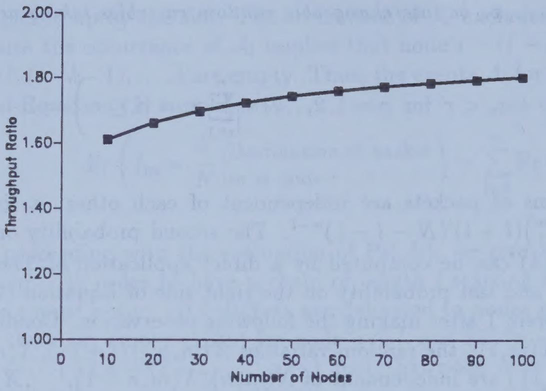


Figure 1: Ratio of the throughput of our model with the throughput of Token Ring vs. the number of nodes N.

that the factor of two gained in throughput due to spatial reuse in not lost because of fairness. This fact is proven in the next section.

4 Asymptotic Analysis

In this section we consider the case in which the number of nodes on the ring tends to infinity. It is shown that the number of circulations made by the slot to evacuate an N-node ring tends to $N/2$, i.e., $\lim_{N\to\infty} 2T_{evac}N^{-1} = 1$. The main result of this section is stated in the following theorem.

Theorem 2 *Consider an N-node ring operating as described in Section 2. Then, the time taken to evacuate the ring is equal to $\frac{1}{2}N + \mathrm{o}(N)$.*[3]

The remainder of this section is devoted to the proof of this theorem. From the previous section, it is known that the time taken to evacuate the ring is given by $\frac{1}{2}(N+1) + \sum_{m=0}^{N-1} E[f_m]$, where f_m is the *free distance* between the destination of the m'th packet and the source of the next packet. Theorem 2 is proven by showing that the sum of the free distance, f_m, for $m = 1, 2, \ldots, N$ is $\mathrm{o}(N)$. The exact probability distribution for the free distance in an N-node ring is given by Equation (5) as

$$\Pr\left(f_m = \frac{k}{N}\right) = \sum_{l=k}^{m} \frac{1}{N^m}\binom{m}{l}(l+1)^{l-1}(N-l-1)^{m-l}\left(1 - \frac{m-l}{N-l-1}\right).$$

Since $(l+1)^{l-1} \le l^l$, for $l > 0$ and $(N-m-1)/(N-l-1) \le 1$, it follows that

$$\Pr\left(f_m = \frac{k}{N}\right) \le \sum_{l=k}^{m} \frac{1}{N^m}\binom{m}{l} l^l (N-l)^{m-l}. \tag{7}$$

---

[3] $f(N) = \mathrm{o}(N)$ implies that $\lim_{N\to\infty} N^{-1}f(N) = 0$.

The expected free distance is given by

$$E\left[f_m\right] = \frac{1}{N}\sum_{k=1}^{m} k \Pr\left(f_m = \frac{k}{N}\right) \leq \frac{1}{N}\left(m_1 + \sum_{k=m_1}^{m} k \Pr\left(f_m = \frac{k}{N}\right)\right), \tag{8}$$

where $m_1 > 0$ is any integer.

Equation (8) can be combined with Equation (7) to give an upper bound on $E[f_m]$ as follows:

$$\begin{aligned} E\left[f_m\right] &\leq \frac{1}{N}\left(m_1 + \sum_{k=m_1}^{m} k \sum_{l=k}^{m} \frac{1}{N^m}\binom{m}{l} l^l \left(N-l\right)^{m-l}\right) \\ &= \frac{1}{N}\left(m_1 + \sum_{l=m_1}^{m} \frac{l(l+1)}{2}\frac{1}{N^m}\binom{m}{l} l^l \left(N-l\right)^{m-l}\right) \end{aligned} \tag{9}$$

where the second equality follows from exchanging the order of summation between k and l and doing the resulting summation over k. The next two lemmas provide the key steps in bounding $E[f_m]$. Their proof, which is omitted here for lack of space, follows from an application of Stirling's formula [12].

Lemma 2

$$\frac{l(l+1)}{2}\frac{1}{N^m}\binom{m}{l} l^l \left(N-l\right)^{m-l} \leq \frac{l(l+1)}{2}\frac{1}{N^{m_2}}\binom{m_2}{l} l^l \left(N-l\right)^{m_2-l} \tag{10}$$

for $0 < m \leq m_2 \leq N$.

Lemma 3

$$\frac{l(l+1)}{2}\frac{1}{N^m}\binom{m}{l} l^l \left(N-l\right)^{m-l} \leq \sqrt{\frac{1}{8\pi}} m^2 \left(\frac{m}{N}\right)^m \left(\frac{N-m_1}{m-m_1}\right)^{m-m_1} \tag{11}$$

for $1 \leq m_1 \leq l < m$.

Lemmas 2 and 3 imply that for $0 \leq m < m_2$ and $m_1 \leq l \leq m$,

$$\begin{aligned} \frac{l(l+1)}{2}\frac{1}{N^m}\binom{m}{l} l^l \left(N-l\right)^{m-l} &\leq \sqrt{\frac{1}{8\pi}} m_2^2 \left(\frac{m_2}{N}\right)^{m_2} \left(\frac{N-m_1}{m_2-m_1}\right)^{m_2-m_1} \\ &\leq N^2 \left(\frac{m_2}{N}\right)^{m_2} \left(\frac{N-m_1}{m_2-m_1}\right)^{m_2-m_1} = A(m_1, m_2). \end{aligned}$$

Equation (9) together with the definition of $A(.,.)$ implies that for $m_2 < N$,

$$\sum_{m=1}^{m_2-1} E[f_m] \leq \sum_{m=1}^{m_2-1} \frac{1}{N}\left(m_1 + (m - m_1 + 1)A(m_1, m_2)\right) \leq m_1 + N^2 A(m_1, m_2). \tag{12}$$

The problem now reduces to finding a bound on $A(m_1, m_2)$. Let $m_1 = N^a$ and $m_2 = N - m_1$, where $0 < a < 1$, and substitute in the expression for $A(m_1, m_2)$ to obtain

$$\begin{aligned} A(N^a, N - N^a) &= N^2 \left(\frac{m_2}{N}\right)^{2m_2 - m_1} \left(\frac{N}{m_2 - m_1}\right)^{m_2 - m_1} \\ &= N^2 \left(1 - N^{a-1}\right)^{N(2-3N^{a-1})} \left(1 - 2N^{a-1}\right)^{-N(1-2N^{a-1})}. \end{aligned}$$

The following lemma, whose proof is also omitted, provides bounds that simplify the previous expression.

Lemma 4 *For* $0 \le x \le \frac{1}{2}$,

$$\exp\left(-x - \frac{1}{2}x^2 - \frac{1}{3}x^3 - \frac{1}{2}x^4\right) \le 1 - x \le \exp\left(-x - \frac{1}{2}x^2 - \frac{1}{3}x^3 - \frac{1}{4}x^4\right).$$

Letting $x = N^{a-1}$, Lemma 4 implies that

$$\begin{aligned} A(Nx, N - Nx) &\le N^2 \exp\left(N(2 - 3x)(-x - \frac{1}{2}x^2 - \frac{1}{3}x^3 - \frac{1}{4}x^4)\right) \times \\ &\qquad \exp\left(-N(1 - 2x)(-2x - \frac{1}{2}4x^2 - \frac{1}{3}8x^3 - \frac{1}{2}16x^4)\right) \\ &= N^2 \exp\left(-Nx^3 \frac{1}{12}(6 - 38x + 183x^2)\right). \end{aligned}$$

Since $183x^2 - 38x + 6 > 4$ for $x > 0$, it follows that

$$A(N^a, N - N^a) \le N^2 \exp\left(-\frac{1}{3}N^{3a-2}\right).$$

Since $E[f_m] \le 1$, $\sum_{m=m_2}^{N-1} E[f_m] \le N - m_2$. This last fact, combined with the substitution of the previous bound on $A(.,.)$ in Equation (12) gives

$$\sum_{m=1}^{N} E[f_m] \le 2N^a + N^4 \exp\left(-\frac{1}{3}N^{3a-2}\right) = O(N^a), \qquad \frac{2}{3} < a < 1.$$

This completes the proof of Theorem 2.

5 Conclusion

In this paper, slotted rings with spatial reuse and a fairness mechanism were considered. We obtain an exact expression for the throughput, and it was shown that the penalty for fairness gets arbitrarily small asymptotically with the number of nodes.

The results show that even for rings with a small number of nodes the penalty for fairness is at most 10 – 20% of the network throughput (see Figure 1). This is a small penalty, in exchange for which the network gains the advantages of fairness. The penalty for fairness can be decreased by increasing the *quota* of each node to more than one. The methods of Section 4 can be extended in a straightforward manner to show that in this case the penalty for fairness also disappears asymptotically with the number of nodes.

Acknowledgements

The authors thank Israel Cidon for bringing this problem to our attention, James Kurose for bringing the Ballot Theorem to our attention, and Hamid Ahmadi for many useful discussions.

References

[1] D. Bertsekas and R. Gallager, *Data Networks.* Englewood Cliffs, NJ: Prentice-Hall, 1987.

[2] F. Ross, "FDDI- a tutorial," *IEEE Communications Magazine*, vol. 24, pp. 10–17, May 1986.

[3] I. Cidon and Y. Ofek, "A Full-Duplex Ring with Fairness and Spatial Reuse." U.S. Patent No. 4926418, 1990.

[4] I. Cidon and Y. Ofek, "MetaRing - a Full-Duplex Ring with Fairness and Spatial Reuse," in *Proceedings of IEEE INFOCOM '90*, (San Francisco, CA), pp. 969 – 981, June 1990.

[5] R. M. Falconer and J. L. Adams, "Orwell: a protocol for an integrated services local network," *British Telecom Journal*, vol. 3, pp. 27–35, October 1985.

[6] H. Muller, et al., "DQMA and CRMA: new access schemes for Gbit/s LANs and MANs," in *Proceedings of IEEE INFOCOM '90*, (San Francisco, CA), pp. 185–191, June 1990.

[7] B. van Arem, "Analysis of a queueing model of the orwell slotted ring," in *Proceedings of Performance'90*, (Edinburgh, U.K.), pp. 61–75, 1990.

[8] M. Zafirovic-Vukotovic, I. G. Niemegeers, and D. S. Valk, "Peformance analysis of slotted ring protocols in HSLAN's," *IEEE Journal on Selected Areas in Communications*, vol. 6, pp. 1011–1024, July 1988.

[9] A. Bhargava, P. Humblet, and M. G. Hluchyj, "Queueing analysis of continuous bit-stream transport in packet networks," in *Proceedings of GLOBECOM '89*, (Dallas, TX), pp. 903–907, IEEE Press, November 1989.

[10] R. W. Wolff, *Stochastic Modeling And The Theory of Queues.* Englewood Cliffs, NJ: Prentice Hall, 1989.

[11] L. Takacs, *Combinatorial Methods in the Theory of Stochastic Processes.* New York: John Wiley, 1967.

[12] W. Feller, *An Introduction to Probability Theory and Its Applications*, vol. one. New York: John Wiley & Sons, 3rd ed., 1968.

On Dynamic Channel Allocation In Cellular/Wireless Networks

Srikanta P. R. Kumar and **Hwan Woo Chung**
Electrical Engineering and Computer Science
Northwestern University
Evanston, Illinois 60208

Mohan Lakshminarayan
Bell Northern Research
Richardson, Texas 75081

Abstract

With the trend towards higher network performance in cellular and wireless networks, traffic sensitive methods that assign available channels to cells (macro and micro) more dynamically (instead of fixed assignment) may prove beneficial. In general, dynamic channel allocation methods can generate better capacity and better handoff performance (lesser forced terminations). Previous studies of dynamic methods are largely simulation based, and theoretical studies aimed at a better understanding of the benefits are relatively few. In this paper, we study several dynamic channel allocation methods, including channel borrowing and hybrid techniques, and analyze them in a stochastic framework. Comparison of allocation methods using stochastic dominance concept is illustrated. This facilitates a finer comparison than the conventional approaches. Bounds indicating the conditions under which dynamic methods considered dominate fixed allocation methods are also developed. Simulation studies are also reported.

1 Introduction

Typical design philosophy in cellular and wireless communication networks is to assign the available channels in a fixed manner to each cell, with spatial reuse. Co-channel interference considerations and the cellular geometry dictate the size and structure of reuse clusters. The conventional approach to increasing capacity is via cell splitting. With the anticipated rapid growth in cellular and wireless network traffic, and the trend towards higher network performance, traffic sensitive methods that assign available channels to cells more dynamically, instead of fixed assignment, may prove beneficial. In general, dynamic channel allocation methods exhibit better call carrying capacity, and also better handoff performance in terms of lesser forced terminations. As some recent studies indicate, dynamic allocation may be necessary in many instances. For example, in microcellular system (high density, interference limited, irregular zones), spatial volatility of traffic is likely to be high, and therefore dynamic methods may be virtually essential.

In this paper, we study several dynamic channel allocation methods, including channel borrowing and hybrid techniques, and analyze them in a stochastic framework. Comparison of allocation methods using stochastic dominance concept is illustrated. This facilitates a finer comparison than the conventional approaches. Bounds indicating the conditions under which dynamic methods considered dominate fixed allocation methods are also developed. Simulation studies are also reported.

The paper is organized as follows. In section 1.1, we briefly discuss some general aspects of dynamic allocation. Section 2 describes the model considered. In section 2.1, we illustrate the use of stochastic dominance in comparing the call carrying capacity of channel borrowing schemes and fixed allocation, ignoring co-channel interference consideration. Section 2.2 extends the analysis to incorporate spatial reuse and interference effects, and to derive sufficient conditions for dynamic methods considered to perform better than fixed allocation. Some simulation results are presented in section 3.

1.1 Dynamic Channel Allocation Strategies

Various dynamic allocation methods in cellular systems may be derived based on a number of considerations. Some of these factors, and a rough classification are given below.

Spatial Boundaries for Channel Sharing : In the simplest case, channel sharing is limited to adjacent cells. In other words, each cell is allowed to borrow or lend a channel only to its neighboring cells. The number of shared channels may also be limited to some fixed quantity. More generally, channel sharing or pooling boundaries may be extended to k-nearest neighbors ($k > 1$), to some specified clusters, or even to the entire area of cellular coverage.

Hybrid Allocation (some fixed, some shared) : Some of the available channels are fixed for each cell (with reuse clusters for these determined from usual considerations), whereas a certain number of channels are available for sharing.

Allocation Dependent on State/Traffic : While the above specify the region for sharing and the number of channel shared, they do not specify when a channel should be shared. This decision could be based on a number of factors. For example, the states or the number of calls in progress (or even some function of carried or offered traffic) in adjacent cells is one factor. More generally, the dynamic allocation could be based on the states of adjacent clusters or the states of some well defined relevant region (in which the impact of the decision will be most felt).

Other considerations may include allocation across user/service classes [9], or allocation in a hierarchical architecture. With macro cells overlaid on micro cells, channel sharing could be restricted to within the same level of hierarchy, or sharing could be permitted across the levels.

2 Network Models and Channel Allocation

In this section, we formulate and analyze simple mathematical models of a cellular network and dynamic channel allocation methods. Particular attention is paid to call carrying capacity; other performance measures such as blocking probability of first-attempt calls and probability of forced termination will be discussed. The notion of stochastic dominance, which is useful in this context is reviewed next. In section 2.1, we illustrate the comparison technique by examining a channel borrowing scheme in a simple example ignoring co-channel interference consideration. In section 2.2, extension to include interference effects is considered.

Definition 1 *A random variable X stochastically dominates another random variable Y, denoted $X \geq_s Y$ if $Pr[X > \alpha] \geq Pr[Y > \alpha]$, for all α.*

In the models considered, the random variables will be the calls carried (or the state at a certain instant or in steady state) in two different systems.

For any revenue function $r(\cdot)$ which is increasing, the above definition is equivalent to the condition

$$E[r(X)] \geq E[r(Y)]. \tag{1}$$

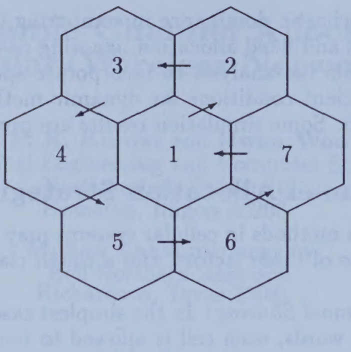


Figure 1: *7-cell structure*

By choosing different revenue functions, different measures of performance can be considered. If the random variable denotes the number of calls carried, then taking r as identity yields the expected call carrying capacity comparison via (1).

In the Markov models of networks considered, the following technique will be used to generate stochastic dominance results. Given two systems $i = 1, 2$, let X_t^i be the state of system i at time t. To show system 1 is better than system 2, we shall assume dominance at time t (i.e., $X_t^1 \geq_s X_t^2$), and show that the network evolution is such that the same dominance holds at time $t+dt$ also. This would, then, imply that given that both systems are started up with the same initial conditions, one system will be stochastically better than the other at each point in time, and also the expected revenues generated will be higher.

2.1 A Simple Illustration

Consider the seven cell system shown in Fig. 1. Suppose there are N distinct channels assigned per cell. Traffic (calls) originates in each cell according to a Poisson process with rate λ. We abstract away from cell design considerations. Assume that each cell is of equal size, and that traffic is spatially homogeneous. Assume there is a single service class, and each call connected (or in progress) hangs up at constant hazard rate μ; and the handoff to an adjacent cell also has a constant hazard rate δ. This implies both conversation and handoff times are exponential, and a call successfully handed off may be viewed as a new call (due to the memoryless property). Assume that the handoff directions are uniform in space (six directions), and the cells are wrapped around. Call set-up and handoff processing times are assumed negligible.

Suppose that a cell can borrow (on demand when all of its current channels are busy) up to k channels from m other cells, in a symmetric pattern. An example pattern for $m = 1$ is shown via the arrows in Fig. 1; cell 1 can borrow from cell 7, cell 2 can borrow from cell 1 and so on. It should be pointed out that the cells need not be hexagonal; they could be rectangular or any other shape.

Let X_t^i denote the number of calls carried at time t for $i = b$(borrowing scheme) or f(fixed allocation). The general idea of the comparison is to consider the time interval $[t, t + dt]$, and first show that, in this interval, the probability of an upward transition is higher and that of a downward transition is lower for the borrowing scheme than for fixed allocation. This property can then be used for characterization via (1). The downward and upward transition probabilities are shown in Fig. 2, for $X_t^i = x$. ($o(dt)$ term has been omitted, as it is harmless.) The borrowing schemes considered generate state transitions

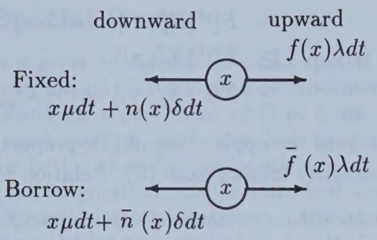


Figure 2: *Transition Probabilities in the Example*

such that

$$\bar{n}(x) \le n(x), \quad \text{and,}$$
$$\bar{f}(x) \ge f(x) \quad \text{for all } x \tag{2}$$

To illustrate (2) for $N = 1$, that is, one channel per cell, consider the borrowing pattern shown in Fig. 1. For fixed assignment, $n(x) = x(\mu + (x-1)\delta)$ for $x = 0, \cdots, 7$. However, for borrowing $\bar{n}(x)$ is equal to $n(x)$ or takes a value less than $n(x)$ which depends on the pattern of the calls in progress, and the borrowing pattern. For example, in the system with $N = 1$, let $x = 3$. Suppose there are calls carried in cells 1, 2 and 7; then $\bar{n}(x) = 3\mu + 2\delta$, since handoffs from cell 2 to cell 1, and from cell 7 to cell 2 will result in forced termination . But for the case when calls are residing in cells 1, 4 and 7, $\bar{n}(x) = 3\mu + \delta$, since the only forced termination from this state is a handoff from cell 4 to cell 1. And finally, for the calls residing in cells 1, 2 and 6, $\bar{n}(x) = 3\mu + \delta$. In all cases, (2) is satisfied.

We wish to point out that even for the non-wrap around structure, (2) holds. With assumptions as above, the following hold. For calls in 1, 2 and 7, $\bar{n}(3) = 3\mu + 2\delta$; for calls in 1, 4 and 7, and for calls in 1, 2 and 6, $\bar{n}(3) = 3\mu$.

Given the relationship (2), it can be established that in this model, borrowing schemes are better than fixed allocation, in the sense explained in previous section. Define for $i = b, f$,

$$Pr[X^i_{t+dt} \ge x \mid X^i_t = y] \equiv t^i(y; x)$$

The following table lists the above transition probabilities.

y:	0	1	2	$\cdots$	$x-1$	x	$x+1$	$\cdots$
$t^f(y;x)$:	0	0	0	$\cdots$	$f(x)\lambda dt$	$[1 - x\mu dt - n(x)\delta dt]$	1	$1\cdots$
$t^b(y;x)$:	0	0	0	$\cdots$	$\bar{f}(x)\lambda dt$	$[1 - x\mu dt - \bar{n}(x)\delta dt]$	1	$1\cdots$

Two properties in the above table are important.

$$(i) \quad t^b(y; x) \ge t^f(y; x), \text{ for all } y \text{ and } x \tag{3}$$
$$(ii) \quad t^i(y; x) \text{ is increasing in } y, \text{ for } i = f, b. \tag{4}$$

Now for every state x,

$$Pr[X^b_{t+dt} \ge x] = \sum_y t^b(y; x) Pr[X^b_t = y]$$

$$\begin{aligned} &= E[t^b(X_t^b;x)] \\ &\geq E[t^b(X_t^f;x)] \\ &\geq E[t^f(X_t^f;x)] = Pr[X_{t+dt}^f \geq x] \end{aligned} \tag{5}$$

The first inequality follows from the application of (1), property (4), and the assumption $X_t^b \geq_s X_t^f$. The second inequality follows from (3). Relation (5) establishes :

Proposition 1 *In the model (of no co-channel interference) above, if the systems with borrowing and fixed channel allocation schemes start in the same state, then for all $t \geq 0$, $X_t^b \geq_s X_t^f$.*

In other words, calls carried is stochastically larger at all times with borrowing scheme than with fixed allocation. The same result holds in steady state (initial state does not matter as both systems are ergodic). Furthermore, for any increasing revenue function, expected revenues are better with the borrowing scheme. It can also be shown that the rate of successful handoffs and connection of first attempt calls will be better. For example, to show handoff completions are better, define the rates $r_b(x) \equiv 6x\delta - \bar{n}\ (x)$ and $r_f(x) \equiv 6x\delta - n(x)$. It is easily verified that, for $x = 0, 1, \cdots, 7$, $r_f(x)$ equals $0, 6\delta, 10\delta, 12\delta, 12\delta, 10\delta, 6\delta$, and 0; and $r_b(x)$ is greater than or equal to $0, 6\delta, 12\delta, 16\delta, 19\delta, 15\delta, 11\delta$, and 7δ. It is, then, clear that there exists an increasing function $r_0(x)$ such that

$$r_b(x) \geq r_0(x) \geq r_f(x). \tag{6}$$

Let C_i be the expected success rate of handoffs for $i = b, f$. Then

$$\begin{aligned} C_b &= \sum_x r_b(n) Pr[X^b = x] \\ &= E[r_b(X^b)] \\ &\geq E[r_0(X^b)] \\ &\geq E[r_0(X^f)] \geq E[r_f(X^f)] = C_f. \end{aligned} \tag{7}$$

The first and the third inequality (7) follows from (6), and the second inequality is due to $X^b \geq_s X^f$ and the increasing property of $r_0(x)$.

With regard to the call completion, let $\bar{C}\ (x) \equiv x\mu$ denote the rate of completion (calls leaving the network after they are served) when there are x calls in progress. Since $\bar{C}\ (x)$ is increasing in x, it follows that, at all times, the expected call completion rate with the borrowing scheme $\bar{C}_b \equiv E[\bar{C}\ (X_t^b)]$ is higher than that for fixed allocation, $\bar{C}_b = E[\bar{C}\ (X_t^f)]$. Since blocking rate of first attempt calls equals the total arrival rate minus the completion rate, it follows that the borrowing scheme has lesser first attempt blocking.

Proposition 2 *In the above model, the success rates of handoffs and first attempts are higher for the borrowing scheme at all times. Furthermore, calls carried increases stochastically in both k and m.*

Similar results can be derived for other borrowing schemes. In fact, these results are related to the well known concept of trunking efficiency. The next section extends this analysis to include co-channel considerations.

2.2 Model with Spatial Reuse

Co-channel interference is a primary consideration in spatial reuse of channels. In this section, we illustrate how co-channel effects can be incorporated into the analysis outlined in section 2.1. Here, we focus on a particular class of dynamic allocation viz.,, hybrid borrowing channel assignment method, and show how one can ascertain conditions for which channel borrowing is better than fixed allocation.

We retain the traffic (call origination, completion, and handoff) assumptions of section 2.1, with N channels per cell. Suppose the reuse cluster size is seven (2-cell buffering); hence the total number of available channels is $7N$.

The hybrid allocation considered is a mixture of fixed allocation and channel borrowing on demand [12][13]. Suppose the N channels of each cell are divided into two sets : a fixed set of channels dedicated to the cell, and a set its adjacent cells can borrow. Let A_i denote fixed set of $N-k$ channels, and B_i denote set of k borrowble channels. Since the reuse cluster size is seven, we need to identify only $B_1, \cdots, B_7$. The method corresponds to a fixed frequency plan, i.e., cell i is designed to handle the frequencies $A_i \cup [\bigcup_{i=1}^{7} B_i]$.

The borrowing strategy is as follows (see [14] and [15]). A cell, on demand, (to handle a call when all channels currently in it are busy) borrows a channel from the B-set of a neighbor. Highest priority is given to a neighbor with maximum B-set availability. Once a channel is borrowed, co-channel cells in the adjacent clusters are locked, i.e., the channel cannot be used in those cells (with seven cell reuse pattern, there are two adjacent clusters that are affected).

A cell i releases a borrowed channel (to its original or home cell), if a channel from its fixed set A_i or its own B_i is available, by switching the call. If there is a choice of which borrowed channel to release, priority may be assigned on the basis of first-in-first-out or the neighbor (home cell) which is most busy. Several variations around this borrowing scheme are possible and have been reported [14][15][16]. The precise details of the borrowing scheme are not necessary in the analysis (to follow) of the condition for which borrowing dominates fixed allocation.

In practice, the above model of a cellular system can be analyzed by conventional Markov chain techniques. However, seeking solution (to find expected call capacity etc.) via this straight forward approach of finding the steady state probabilities of the Markov chain (numerically or via simulation) could become an unwieldy task, especially in large networks (large number of clusters, channels etc.), due to very large number of states needed to characterize the system.

We examine, in the following, conditions under which borrowing schemes do better than fixed allocation. Our starting point is a single reuse cluster considered in section 2.1; to this, we attempt to incorporate factors reflecting the co-channel effects of borrowing in a way that enables the comparison of the allocation methods. As we wish to keep the extension simple, the analysis presented is approximate (refinements are possible), and the bounds derived are conservative.

The approach taken here is to examine the positive and negative effects of borrowing a channel. Consider the moment at which cell i wants to borrow a channel from cell j. One positive aspect to borrowing the channel is the completion of this call at hand in cell i. There are two negative factors : first, a potential (future) call in cell j may be lost due to channel lending; and second, potential calls in the co-channel cells (say, N_c in number) in the clusters adjacent to that of cell i, may be lost due to locking. The main implication of the analysis of section 2.1 is that, in the absence of co-channel interference, the net effect to borrowing (positive minus the first negative effect) is positive. Taking the somewhat simplified picture of all the effects as described above, the expected benefits from a single call on a borrowed channel, denoted V may be determined as follows.

Suppose cell i wants to borrow from cell j at time t. Let k_0 represents the number of busy channels in cell i at this instant. Considering a short time interval $[t, t+dt]$, the probability a call completion occurs (approximately, the probability of returning this borrowed channel) is $(k_0+1)\mu dt$. This may be viewed as generating a unit reward. A handoff from cell i to cell j in this interval, which has probability $(k_0+1)\delta dt$, may be seen as resulting in the return of the borrowed channel. A handoff to any other cell $k(\neq j)$, which occurs with probability $(k_0+1)5\delta dt$, may or may not result in borrowing by cell k. It is conservative to assume that it does, and we do so. In this case, the expected benefits continues to be V. In the event, the status of the borrowed channel in cell i does not change (see third term on the right side of (8) below), no gains are accrued during dt, and the expected future benefits remain at V. On the negative side, the expected penalty paid during this interval is $N_c\lambda dtP$, where P denotes the probability of attempt in one of the N_c co-channels in the adjacent clusters. Hence

$$V=(k_0+1)(\mu+\delta)dt+(k_0+1)5\delta dtV+[1-(k_0+1)(\mu+6\delta)dt]V-N_c\lambda dtP+o(dt) \quad (8)$$

Manipulation of (8) yields

$$V=1-N_c\lambda P/(k_0+1)(\mu+\delta). \quad (9)$$

If $V>0$, borrowing a channel generates a positive revenue flow. The entire cellular system may now be viewed clockwise, by taking appropriate values for the parameters in (9). In particular, P and k_0 need to be determined. Assessing their values precisely in a system with large number of cells (from the Markov model of the entire network) is involved. A simple approximation is as follows. A conservative value of P is $E_r(\lambda/\mu; N-k)$, where E_r is the Erlang-B formula with traffic intensity $\lambda/\mu\equiv\rho$, and $N-k$ available circuits; this assumes that if all fixed channels in a co-channel cell are busy, then an arriving call there generates an attempt on the (borrowed) channel under consideration. If on an average k_0 is taken to be N, then $V>0$ translates to

$$\frac{N_c\lambda E_r(\rho;N-k)}{(N+1)(\mu+\delta)}<1. \quad (10)$$

The analysis in section 2.1, can be extended to show that if $V>0$, then expected calls carried with dynamic allocation is better than fixed allocation. The formal proof is omitted here. (The idea is to account each cell on a own channel of cell as generating a unit revenue flow, and each borrowed call as having a revenue flow of V.)

Conditions (9) and (10) give estimates of when channel borrowing methods considered is better than fixed allocation. ((10) gives the conditions directly in terms of the model parameters.) The technique used in arriving at these estimates is to account the benefits and penalties of borrowing, and then to incorporate this in the analysis of a single cluster outlined in section 2.1. The same technique could be used to analyze other dynamic methods as well. The next section presents simulation results.

3 Simulation

The system used in the simulation study consists of 49 cells. The number of cells per cluster is chosen to be seven. The call arrivals in each cell are assumed Poisson, and the rate is same for all cells. The conversation time is exponential with average of 98 seconds. Both no-handoff and handoff cases are considered. In the latter, 10% of traffic on average move across cell boundaries, and the call residing time in a cell is exponential with average 88.2 seconds. Handoffs are uniform in all directions. The number of channels per cell is 10. The figure of 5 Erlangs (which yields 0.018 blocking probability for fixed assignment) was used as the starting point of the simulation. The borrowing method simulated is as described in section 2.2. For borrowing, the neighboring cells are scanned clockwise, starting from the north neighbor. A channel is borrowed from a neighbor with maximum B-set availability. In case of a tie, the one which appears first in the selection

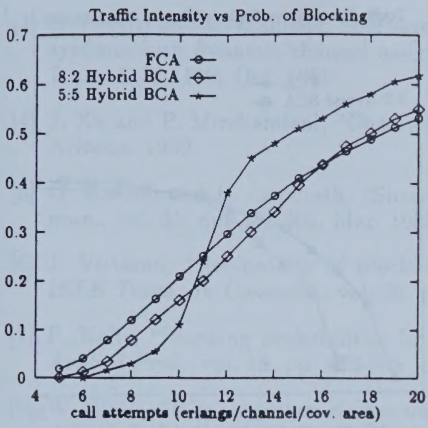


Figure 3: *Traffic Intensity vs. Blocking Probability of First Attempts (no-handoffs)*

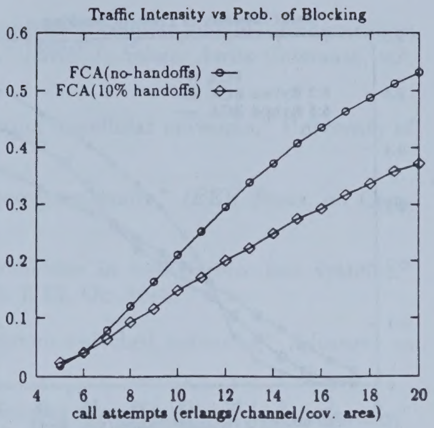


Figure 4: *Traffic Intensity vs. Blocking Probability for FCA (10% handoffs and no-handoffs)*

with FCA	no-handoffs		10% handoffs	
	8:2 BCA	5:5 BCA	8:2 BCA	5:5 BCA
Theoretical (Erlangs)	15.92	10.55	15.11	9.96
Simulation (Erlangs)	15.80	11.00	15.20	10.60

Table 1: *Cut-off points for blocking probabilities*

order is selected. One of the borrowable channels in the cell selected is used to carry a new call. During the duration of the call, that particular channel in the co-channel cells is locked or marked as unavailable. Normal calls which use fixed channels in one cell are terminated as soon as the holding time of a channel expires in the normal pattern. But calls which use borrowable channels in their cells or borrowed channels from the neighbor cells switch to a fixed channel as soon as one is available. In the latter, in addition to the switching to a fixed channel, the channel is unlocked from the relevant cells. Calls are terminated as soon as the holding time expires, with unlocking of the channels if they used borrowed channels. Assignment procedures are same for new and handoff calls. A blocked handoff call is a forced termination.

3.1 Simulation Results

The results of simulations are shown in Fig. 3 – Fig. 6. These present the probability of first attempts as a function of the traffic intensity $\rho = \lambda/\mu$ (erlangs/channel/cell coverage area). Fig. 6 presents the probability of forced terminations. The dynamic methods perform better than fixed allocation in the low traffic region as expected (see equations (9) and (10), in terms of both blocking probability of first attempts and probability

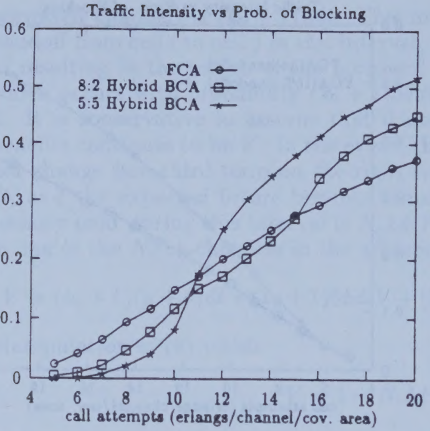


Figure 5: *Traffic Intensity vs. Blocking Probability of First Attempts (10% hand-offs)*

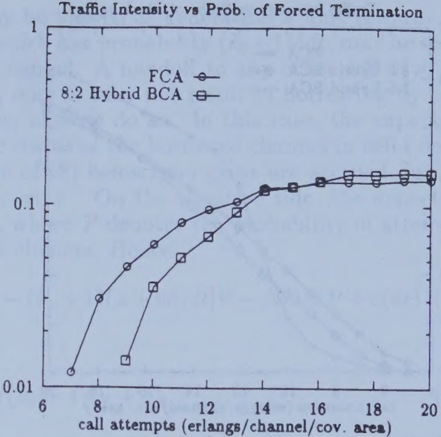


Figure 6: *Traffic Intensity vs. Probability of Forced Termination (10% hand-offs)*

of forced termination. Table 1 presents the cutoff values at which fixed and dynamic allocation curves crossover. There is close agreement between theoretical (from (10)) and simulation values. However, it should be noted that approximations were made in deriving (10); further study is needed to ascertain whether this is a good estimate for all parameter values of the model.

4 Conclusion

In this paper, we have presented an analysis of channel allocation methods, using the notion of stochastic dominance for comparison of fixed and dynamic methods. Estimates of the parameters for which dynamic methods are better than fixed allocation were presented. Some approximations were made in the analysis in section 2.2. This could be made more accurate by incorporating the effects of interference and specific channel allocation procedures features more precisely (particularly for P and k_0 in equation (9)). Issues of incorporating into the model irregular cell geometries, specific propagation models (multipath, fading, shadowing), dynamic allocation of base station and channels based on interference measurements, and spatially nonuniform traffic etc. are important topics for future research.

References

[1] D. Cox and D. Reudink, "Increasing channel occupancy in large-scale mobile radio systems: dynamic channel reassignment," *IEEE Trans. on Veh. Tech.*, vol. VT-22, pp. 218-222, Nov. 1973.

[2] L. Schiff, "Traffic capacity of three-types of common user mobile radio communication systems," *IEEE Trans. on Commun. Tech.*, vol. COM-18, pp. 12-21, Feb. 1970.

[3] D. Everitt and D. Manfield, "Performance analysis of cellular mobile communication systems with dynamic channel assignment," *IEEE J. Select. Areas Commun.*, vol. 7, pp. 1172-1180, Oct. 1989.

[4] J. Xu and P. Mirchandani, "Channel allocation in cellular networks," University of Arizona, 1989.

[5] G. Foschni and B. Gopinath, "Sharing memory optimally," *IEEE Trans. on Commun.*, vol. 31, pp. 352-360, Mar. 1983.

[6] J. Virtamo, "Reciprocity of blocking probabilities in multiservice loss systems," *IEEE Trans. on Commun.*, vol. 36, pp. 1174-1175, Oct.1988.

[7] F. Kelly, "Blocking probabilities in large circuit-switched networks," *Advances in Applied Prob.*, vol. 18, pp. 473-505, 1986.

[8] W. Whitt, "Blocking when service is required from several facilities simultaneously," *AT&T Tech. J.*, vol. 64, pp. 1807-1856, 1985.

[9] S. Jordan and P. Varaiya, "Throughput in multiple service, multiple resource communication networks," to appear in *IEEE Trans. on Commun.*, 1990.

[10] G. Choudhury and S. Rappaport, "Cellular communication schemes using generalized fixed channel assignment and collision type request channels," *IEEE Trans. on Veh. Tech.*, vol. VT-31, pp. 53-65, May 1982.

[11] D. Hong and S. Rappaport, "Priority oriented channel access for cellular systems serving vehicular and portable radio telephones," *IEE Proc.*, vol. 36, pt. I, pp. 339-346, Oct. 1989.

[12] J. S. Engel and M. M. Peritsky, "Statistically-optimum dynamic server assignment in systems with interfering servers," *IEEE Trans. Veh. Tech.*, vol. VT-22, no. 4, pp. 203-209, Nov. 1973.

[13] T. J. Kahwa and N. D. Georganas, "A hybrid channel assignment scheme in large-scale, cellular-structured mobile communication systems," *IEEE Trans. Commun.*, vol. COM-26, no. 4, pp. 432-438, Apr. 1978.

[14] S. M. Elnoubi, R. Singh, and S. C. Gupta, "A new frequency channel assignment algorithm in high capacity mobile communication systems," *IEEE Trans. Veh. Tech.*, vol. VT-31, no. 3, pp. 125-131, Aug. 1982.

[15] M. Zhang and T. P. Yum, "Comparisons of channel-assignment strategies in cellular mobile telephone systems," *IEEE Veh. Tech.*, vol. 38, no. 4, pp. 211-215, Nov. 1989.

[16] L. G. Anderson, "A simulation study of some dynamic channel assignment algorithm in a high capacity mobile telecommunication system," *IEEE Trans. Veh. Tech.*, vol. VT-22, no. 4, pp. 210-217, Nov. 1973.

[17] S. Nanda and D. J. Goodman, "Dynamic resource acquisition : A scheme for carrier allocation in cellular systems," *Proc. of Third Gen. Wireless Info. Networks*, Rutgers, NJ, Oct. 1990.

[18] S. Ross, *Introduction to stochastic dynamic programming*, Academic Press, New York, NY, 1983.

Efficient Algorithms For Shared Memory Consensus

Karl Abrahamson and Ashish Karkare

Email:[karl,akarkare]@yoda.eecs.wsu.edu
Department of Computer Science
Washington State University
Pullman, WA 99164-2752

Abstract

A *consensus* algorithm allows all the non-faulty processes in a system of n processes to reach a common decision value that is the initial value of some process. The consensus problem has been studied in several different models in both message passing and shared memory based distributed systems. We address the consensus problem in shared memory systems.

We present two randomized consensus algorithms, which use *multi-reader, multi-writer* registers. Both algorithms have an expected running time of $\mathcal{O}(n)$. The first algorithm assumes an "oblivious" scheduler similar to the message oblivious scheduler described by Chor *et al* [CMS89] that does not see the contents of either the shared memory or the local memory of any process. The second algorithm works against an "omnipotent" scheduler provided that a randomized-write similar to the one used by Abrahamson [A88] is available as an atomic action. Both algorithms are wait-free and have modest expected space requirements.

Our algorithms are at least an order of magnitude faster than the previously known algorithms presented in [A88], [HA88], and [ADS89]. They are simple enough to be practicable, and powerful enough to achieve termination in most realistic situations.

1. Introduction

In this paper we present *wait-free* algorithms for consensus in asynchronous, shared memory systems where up to $n-1$ processes may fail only by stopping. A *consensus* algorithm allows all the live processes in the system of n processes $P_1 \cdots P_n$, each starting with input 0 or 1, to reach a common decision value that is the initial value of some process. *Consistency* requires that all processes decide the same value. *Termination* requires that each process should complete execution of the algorithm in a bounded number of steps. The maximal number of computation steps taken by a given process during the execution of the algorithm, the *time complexity*, and the maximal amount of shared memory required for the execution of the algorithm, the *space complexity*, are two quantitative measures for evaluating a consensus algorithm. The "strength" of the model serves as a qualitative measure for evaluating the algorithm.

Consensus is a powerful tool with wide applications in distributed computing. Application examples are: wait-free solutions to *distributed decision tasks* [CM89]; implementation of universal primitives for wait-free synchronization such as *fetch-and-cons* [H88] and *Sticky Bit* [P89]; distributed *commit*; and multivalued agreement. The practicability of these applications, however, depends on how efficiently one can solve the consensus problem. Thus, efficient solutions to the consensus problem are highly desirable.

It has been shown by Louis and Abu-Amara [LA87] and also by Chor, Israeli and Li [CIL87] that no deterministic algorithm can solve asynchronous, wait-free, shared memory consensus. Randomized solutions for the problem have been presented in [CIL87, A88, HA88, ADS89]. Although these algorithms overcome an omnipotent scheduler, even a simple lock-step strategy brings out a fairly bad case behavior, indicating that these algorithms do not perform significantly better against weaker schedulers.

It is well known that the power attributed to the adversary scheduler has a profound effect on the design of distributed algorithms. In principle, it is possible for a simple algorithm to overcome a weaker adversary. In practice, it has been frequently observed that given the option between a simple algorithm that will work in most realistic situations, and an intricate algorithm that never errs, system designers will very often prefer the former to the latter, especially if the simple algorithm is considerably more efficient. Choosing the more efficient algorithm does not necessitate abandonment of correctness proofs. It suffices that one is aware of the exact limitations of the algorithm.

We illustrate the above principle by presenting simple and efficient algorithms for asynchronous shared memory consensus. These algorithms can be easily implemented on existing shared memory architectures. In order to improve the efficiency, we have placed some restrictions on the scheduler, and these restrictions might seem reasonable in most cases. In this paper we characterize the models in which these algorithms work and also prove their correctness within the scope of these models.

Most known consensus algorithms demand a *snapshot* of a designated set of shared registers in each round to obtain a consistent view of the shared memory and this considerably increases the cost of the algorithm. We seek to either eliminate the snapshot, or if that isn't feasible, reduce the cost of the operation.

Our first result, algorithm $\mathcal{A}$, is an $\mathcal{O}(n)$ algorithm that works against an "oblivious" scheduler. The novel feature of this algorithm is that it needs to take a snapshot that involves only a few registers, thereby reducing the cost of the snapshot. Another feature is that it can be made to run in $\mathcal{O}(m)$ time when all but m of the processes are initially dead. This improvement can be made by incorporating a strategy similar to binary exponential back off into the algorithm. Our second algorithm also has linear expected time and works against an "omnipotent" scheduler provided that a randomized write is available as a primitive operation. In the absence of such a primitive, this algorithm works against a slightly weaker scheduler. Both these algorithms are a significant improvement over previously known wait-free consensus algorithms.

The outline of the rest of the paper is as follows. Section 2 briefly discusses our models for shared memory systems. This discussion is necessary to relate our results to the known results on shared memory consensus. In Section 3, we present a consensus algorithm which achieves termination given an oblivious scheduler. In Section 4, we extend the same approach to achieve termination against an omnipotent scheduler. Section 5 summarizes our results and outlines the scope for future research.

2. Model

Our model for shared memory systems is based on the ones described in [LA87] and [CIL87]. Some changes have been made to make the model more suitable for our needs. Any execution of the system is viewed as an interleaving of the *atomic steps* of individual processes, which simplifies the analysis of algorithms. We define an atomic step to be either a local coin toss operation, or a shared memory access combined with local *deter-*

ministic computation. Multiple processes may access the same memory location for both reading and writing.

However, if tossing a coin and a write to the shared memory are performed as a single atomic step, a significant speedup can apparently be achieved, as demonstrated in [A88]. The randomized write of [A88] is similar to the probabilistic transition function of [CIL87]. In a randomized write, a coin is tossed first. If the outcome is heads (H), a write is performed. The operation *r-write (p, reg, value)* has three parameters: p, which is the probability of success; *reg*, which is the target of the write; and *value*, which is the value written to *reg*.

We define a *scheduler* to be an agent who decides the next process to take a step on the basis of the knowledge available to the scheduler. The role of the scheduler is to factor out nondeterministic choices in the system. A scheduler can be characterized by the knowledge available to it, which also limits its capabilities. The component of the system state s available to the scheduler at any point in a schedule corresponds to its dynamic knowledge. The scheduler can be viewed as a mapping from $\pi(S) \times N \times V$ to the interval $[1, n]$, where $\pi(S)$ is some projection of the system state, N is the set of natural numbers, and V is the input vector. This gives rise to the idea that restrictions of the mapping that defines the scheduler results in a hierarchy of schedulers that differ significantly in their powers. One can see at least three kinds of schedulers, with decreasing capabilities, viz,

- an "omnipotent" scheduler that sees the entire system state.
- a scheduler that can see the contents of the shared memory only.
- an "oblivious" scheduler that either does not see, or does not care about the contents of the local and shared memories.

If any scheduler can lead an algorithm into an inconsistent decision state, then all schedulers can do so. The choice of the scheduler will only affect the termination of the algorithm.

The model of computation is characterized by the following:

1. The knowledge available to the scheduler.
2. The nature of atomic operations that are permitted.

In Section 3, we develop an algorithm in a model that uses only atomic reads and writes to the shared memory, but denies the scheduler access to the contents of the shared or the local memory. This *oblivious* scheduler is fairly weak and is similar to the message oblivious scheduler of [CMS89]. This model also closely resembles the asynchronous PRAM model of Martel and Subramonian [MS90] due to a severe restriction on the capability of the scheduler to make dynamic decisions.

The capabilities of this restricted scheduler when confronted with a *deterministic* algorithm are not diminished. Given the initial values and a deterministic algorithm, this scheduler can compute the state of any process at any point in a schedule. Therefore, the impossibility results of [LA87] and [CIL87] for deterministic consensus algorithms are applicable to an oblivious scheduler.

The second model that we consider allows the scheduler to be omnipotent, and the repertoire of instructions is supplemented by an atomic randomized write operation. An algorithm based on atomic randomized writes is presented in Section 4. Our randomized write is different from the one used in [A88]. For convenience, we allow the operation to return *SUCCESS* or *FAIL*.

However, it may not be feasible to furnish an atomic implementation of a randomized write. Our algorithm uses an atomic randomized write only to hide the outcome of local coin tosses from the scheduler. This can be achieved easily against a weaker scheduler that does not have access to the contents of the local memory. Against such a scheduler, a software implementation of the randomized write will also be effectively atomic because the scheduler can learn about the outcome of the coin toss only after letting the process take an additional step.

3. Algorithm $\mathcal{A}$

The novel feature of Algorithm $\mathcal{A}$ is that the algorithm needs to take a snapshot that involves only a few registers, which is naturally cheaper than an n register snapshot. This is achieved by our transformation of a n process consensus problem to a C process consensus problem, where $C << n$. An obvious choice of C is one, since one process consensus is trivial and the snapshot consists of reading just a single register. Thus, although this technique works for arbitrary $C > 0$, we give the algorithm for $C = 1$. The technique used for the n to C transformation works only against the restricted scheduler. The algorithm and the *get_lid* function are shown in Figure 1. Note that consecutive local actions are assigned just a single line number.

In Algorithm $\mathcal{A}$, the computation proceeds in a series of rounds. Each iteration of the loop at line 4 is a round, and each round uses a fresh set of registers. We assume that the shared memory is initialized to a $\perp$. In the first round processes start off by choosing an initial value, which is their preferred decision value for that round.

The *while* loop at line 5 is a *delay* loop. In this loop, each process keeps tossing a biased coin, which gives an H with a probability p and a T with probability $1-p$. The bias p is $\frac{1}{cn}$, where c is the cost of lines $6-13$. A process exits its delay loop only if the outcome of the coin toss is an H. The purpose of this loop is to increase the randomization in the algorithm. This makes it futile for the oblivious scheduler to try to guess the state of the process.

After exiting the loop, if consensus has not been reached as yet, the process tries to write to d^r. Before doing so, it has to acquire a valid *logical identifier (lid)* in the range $[1, C]$ by invoking the *get_lid* function. When $C = 1$, at most 1 process can get an *lid* in any round r.

Other processes invoke *new_pref*, which can lead to the aborting of the current round of computation. If some process invoking new_pref succeeds in writing the abort register before any other process writes to d^r, then no process can terminate in that round. Termination of the algorithm is guaranteed when some process completes both its delay section and the 1 process consensus algorithm before any other process exits its delay loop. An outline of the proof of correctness of algorithm $\mathcal{A}$ is given below.

Lemma 1: Algorithm $\mathcal{A}$ satisfies the validity condition.
The reader can easily verify the correctness of this lemma.

Lemma 2: At most 1 process can get a valid logical identifier in any round r, when $C = 1$.

```
$(\forall r, j := 1, ..., C)\ own_0^{j,r} := own_1^{j,r} := abort^r := d^r := \perp$
$v_i \in \{0,1\}, r := 1, p = coin\ bias$
procedure Consensus ( n )
begin Consensus
loop
    while ( flip(p) = T) skip;  /*Random Delay */
    if (decision $= \perp$ ) then
        lid$_i$ := get_lid (i, r) ;
        if (lid$_i \neq FAIL$ ) then
            $d^r := v_i$ ;  /*1 process consensus */
            if ( abort$^r \neq 1$ ) then
                decision := $v_i$ ;
                return $v_i$
            fi
        else
            if ( new_pref () = 1) then
                return $v_i$;
            fi
        fi
        $r := r + 1$
    else
        return  decision
    fi
end loop
end Consensus
```

Figure 1: The Consensus Algorithm $\mathcal{A}$ for P_i

Proof: As we mentioned earlier, processes try to obtain a valid *lid* by executing the *get_lid* routine, which returns either a valid *lid* or a special constant *FAIL*. Consider two processes P_i and P_j executing the *get_lid* routine concurrently. A process can obtain a valid *lid* if and only if it can successfully establish the ownership of a register pair. The criterion for ownership is that the process should write $own_0^{j,R}$, write $own_1^{j,r}$ and after successfully accomplishing this, it should find on reading $own_0^{j,r}$, that the value that it had written has not been overwritten. Any process can write to $own_0^{j,r}$, but it can write to $own_1^{j,r}$ only if $own_1^{j,r}$ contains $\perp$.

At line 30, P_i and P_j choose values k and l respectively. If $k \neq l$, then there is no conflict between P_i and P_j. The only case of interest is if $k = l$, where both processes are trying to own the same register pair.

If P_i does succeed in owning the register pair, then the read by P_i at line 34 precedes the write by P_j at line 31. This implies that when P_j reads $own_1^{l,r}$ at line 32, it will have a value i. This will cause P_j to return *FAIL*. Thus, the transformation is consistent. □

```
17. function new_pref ()
18. begin new_pref
19.   if ($d^r \neq \perp$ )then
20.       $v_i$ := $d^r$;
21.       if (abort$^r \neq 1$)then
              $v_i$ := $d^r$ ;
              return 1
          fi
      else
22.       abort$^r$ := 1;
23.       if ( $d^r \neq \perp$ )then
              $v_i$ := $d^r$ ;
          fi
      fi
26.   return 0
27. end new_pref

28.   function get_lid ( $i$ , $r$ )
29.   begin get_lid
30.   lid := Choose (1,$C$);
31.   own$_0^{lid,r}$ := $i$ ;
32.   if ( own$_1^{lid,r}$ = $\perp$) then
33.       own$_1^{lid,r}$ := $i$ ;
34.       if ( own$_0^{lid,r}$ = $i$)then
                  return $lid$
          fi
      fi
35.   return $FAIL$ ;
36.   end get_lid ;
```

Figure 2: Functions invoked by $\mathcal{A}$

Lemma 3: If some process terminates in a round r with a decision value x, then no process can ever decide on a value other than x.

It is clear that when processes advance from round r to round $r+1$, it will be only because some process writes the *abort* register for round r.

The processes that wish to abort computation in a given round r invoke the *new_pref* routine to choose a new preferred value. In *new_pref* processes take one or more snapshots of the registers used for the C process consensus, and read, and possibly write, the abort register before moving to the next round. Let P_i be a process that completes execution of *new_pref* before any other process finishes its computation in the current round. Figure 3 illustrates the possible cases for process P_i, characterized by the nature of the two snapshots that it takes. D indicates a decisive snapshot and I indicates an indecisive snapshot. There are only three possible cases since a decisive configuration cannot be changed into an indecisive configuration.

Let P_j, P_k, P_l represent processes other than P_i that also execute the *new_pref* routine. The configurations seen by these representative processes are shown in Figure 4. The scenarios shown represent all the possible ways in which processes that invoke *new_pref* can advance to the next round. Now we show that the algorithm preserves consistency in each of these scenarios.

Case 1:

In this case, since the snapshot at line 23 indicates an indecisive configuration, process P_i finishes writing *abort*^{r} before any process finishes the C process consensus. Thus, the write of *abort*^{r} at line 22 by P_i precedes the reading of this register by any process, either at line 10, or at line 21. P_i does not terminate in the current round. Since the write to *abort*^{r} precedes its read by any process, no process that reads *abort*^{r} can terminate

	snapshot 1	snapshot 2
case 1	I	I
case 2	I	D
case 3	D	-

Figure 3: Scenarios for P_i

in the current round. The only processes that read $abort^r$ are P_k, P_k and the processes participating in the C process consensus. It follows that no process can terminate in the round r in case 1.

Case 2:

In this scenario there is a possibility that the read of $abort^r$ at lines 10 or 21 precedes the write by P_i at line 22. Thus, some participant in the C process consensus might terminate. Also, some process P_l might terminate. Let x be the decision value of the processes that terminate. This implies that when P_i takes a subsequent snapshot, it will find an $x-$decisive configuration. At this point it changes its preferred value to x. Since there is no process that prefers a value other than x in a higher round, the validity of algorithm $\mathcal{A}$ guarantees that the eventual decision value is x.

Case 3:

If P_i finds that the abort register has not been set, it will decide on x, if it gets a snapshot of an x-decisive configuration. Otherwise it will adopt x as its preferred value.

As in the case above, all processes advancing to the next round will have the same preferred value. Hence, the common preferred value will be the eventual decision value.

Since the consistency requirement is satisfied in all the cases above, we conclude that algorithm $\mathcal{A}$ is consistent. □

Theorem 4: Algorithm $\mathcal{A}$ will terminate against an oblivious adversary scheduler in a constant expected number of rounds, and the total expected time is $\mathcal{O}(n)$.

Proof Sketch:

Let S_k be an arbitrary schedule consisting of k steps, where $k = cn$, and $c > 0$ is a constant to be determined below. We first consider the probability of the algorithm terminating in the first round in schedule S_k. Let $Cost$ denote the cost of lines 6-11, which are executed before termination. Note that k is a parameter pertinent only to the analysis of the algorithm. Let the probability p of tossing an H be $p = \frac{1}{nCost}$. Let s_i denote the number of steps executed by P_i in schedule S_k. P_i can terminate in S_k if $s_i - Cost > 0$, P_i tosses an H in its first $s_i - Cost$ steps, and no other process does so. The probability of the two events is given by $1 - t^{s_i - Cost}$ and t^{k-Cost} respectively, where $t = 1 - \frac{1}{nCost}$ is the probability of tossing a T. Thus, the probability that some process will decide after the system executes S_k is given by

$$f(s_1, .., s_n) = \sum_{i=1}^{n}(1 - t^{s_i - Cost})t^{k-s_i}$$

The aim of the scheduler is to prevent or delay the algorithm from terminating. Thus, the scheduler chooses the worst possible schedule S_k for the algorithm. This is done by minimizing the probability that only one process exits the delay loop, which is the key

Case 1	P_i	P_j	P_k	P_l
snapshot 1	I	I	I	D
snapshot 2	I	I	D	-

Case 2	P_i	P_j	P_k	P_l
snapshot 1	I	-	I	D
snapshot 2	D	-	D	-

Case 3	P_i	P_j	P_k	P_l
snapshot 1	D	-	I	D
snapshot 2	-	-	D	-

Figure 4: Scenarios for processes invoking *new_pref*

to the termination. This is equivalent to minimizing $f(s_1, .., s_n)$ subject to the constraint on s_i

$$\sum_{i=1}^{n} s_i = k$$

Applying the Lagrange Multiplier technique, we obtain the minimum of f when $s_i = k/n$, i.e. when all processes execute the same number of steps. Thus, the probability that some process decides in the first round under schedule S_k is

$$\mathcal{P}(n) = \sum_{i=1}^{n} (1 - t^{k/n - Cost}) t^{k - k/n}$$

What is the expected cost of the first round? The expected number of steps before a process can exit its delay section is $\mathcal{O}(1/(1-t))$. Since the remaining cost of the round is $\mathcal{O}(1)$, $1/(1-t)$ is the expected cost of round 1.

We simplify analysis for the subsequent rounds by revealing more information to the scheduler. We give the scheduler the ability to detect the transition of any process from one round to the next. This enables the scheduler to block a faster process until all others complete execution of the same round. Doing this, in fact, is to the scheduler's advantage, since all processes are in a known state at the beginning of a round. The analysis is simplified as all rounds look exactly the same. It follows that the expected number of rounds is $1/\mathcal{P}(n)$, and the expected cost of the algorithm is $\frac{1}{\mathcal{P}(n)(1-t)}$. A suitable value for k is $2n\,Cost$. This is because the expected number of steps before some process tosses an H is $n\,Cost$. Since all processes are run in lockstep, and all processes are equally likely to toss an H, each process has to execute at least $2\,Cost$ steps before some process terminates. It also turns out that the probability of success of the algorithm is maximized for the chosen values of k and p.

It follows that $\mathcal{P}(n)$ is maximum when $k = 2n\,Cost$ and $t = 1 - \frac{1}{n\,Cost}$, giving $\mathcal{P}(n) \approx e^{-2}$. Thus, the expected number of rounds is e^2 and the expected cost of the algorithm is $e^2 n\,Cost$, which is $\mathcal{O}(n)$.

It follows that $\mathcal{A}$ terminates in a constant expected number of rounds against the oblivious scheduler. □

4. Algorithm $\mathcal{B}$

In this section we briefly describe our second algorithm that achieves termination against a stronger scheduler. This is done by refining some the ideas (such as, avoiding snapshots, increasing randomization *etc.*) developed in the previous section. This algorithm uses multi reader/writer registers and can tolerate the failure of up to $n-1$ processes.

Algorithm $\mathcal{B}$ uses a very simple idea to avoid an n register snapshot: Process P_i does not need to read the register belonging to process P_j, if it knows that P_j is guaranteed to prefer the same decision value as itself. We partition n processes into 2 groups, where each group consists of $n/2$ processes. We ensure that all the processes in a group prefer the same decision value in any round by first forcing processes within a group to reach a consensus on their preferred value. Following this, it is clear that 2 registers suffice to communicate the preferred values of the two groups to each other, and an n register snapshot is unnecessary.

Randomization is introduced in algorithm $\mathcal{B}$ through the randomized write operation. After the $n/2$ process consensus within a group, by recursively calling the same algorithm, each process enters a loop in which it alternately attempts a randomized write to its group register and reads the register belonging to its complementary group.

The algorithm terminates in a given round if just one group succeeds in writing to its group register. It can be shown that this occurs with a fairly high probability. This leads to an algorithm whose expected cost is $\mathcal{O}(n^{1+\epsilon})$, for $\epsilon > 0$. A simple modification in $\mathcal{B}$ leads to algorithm $\mathcal{B}'$, which is an $\mathcal{O}(n)$ algorithm.

Algorithm $\mathcal{B}$ has a fairly simple structure, that of a two process consensus algorithm. Unfortunately, this structural simplicity leads to a fairly complex proof of correctness, which we present in the full paper.

5. Conclusion

Since its introduction by Pease, Shostak, and Lamport [PSL80] the consensus or the agreement problem has been studied in several different models of distributed computation. Interest in the problem has persisted over a decade because consensus on a single bit underlies problems involving co-ordinated decisions by a set of processes. The models in which the problem has been studied differ in assumptions regarding synchrony, nature of process failures, and the capabilities of the scheduler.

In this extended abstract, we have presented an outline of our new solutions to the asynchronous shared memory version of the consensus problem. Our results illustrate the principle that simple algorithms can overcome weaker adversaries. In retrospect, our first result viz. a $\mathcal{O}(n)$ consensus algorithm against an "oblivious" scheduler, is not surprising as this scheduler is very similar to the non-adaptive scheduler of [MS90]. Our second result, an $\mathcal{O}(n)$ algorithm using randomized-writes is significant as it a substantial improvement over the fastest previously known such algorithm.

The algorithms that we have presented have a modest expected space requirement. However, it is still of academic interest to see if bounded memory versions of these algorithms exist. Another question of interest is whether there is an algorithm along the same lines that works against an omnipotent scheduler. Finally, the question regarding a lower bound for shared memory consensus remains still open.

References

[A88] K. Abrahamson. On Achieving Consensus Using a Shared Memory. In *Proceedings 7th Annual ACM Symposium on Principles of Distributed Computing*,pages 291-302, 1988.

[ADS89] H. Attiya, D. Dolev, N. Shavit. Bounded Polynomial Randomized Consensus. In *Proceedings 8th Annual ACM Symposium on Principles of Distributed Computing*,pages 281-294, 1989.

[BT85] G. Bracha, S. Toueg. Asynchronous Consensus and Broadcast Protocols. *J. ACM,* 32(4):pp. 824-840, 1985.

[CIL87] B. Chor, A. Israeli, M. Li. On Processor Coordination Using Asynchronous Hardware. In *Proceedings 6th Annual ACM Symposium on Principles of Distributed Computing*,pages 86-97, 1987.

[CM89] B. Chor, L. Moscovici. Solvability in Asynchronous Environments. In *Proceedings, IEEE 30th Annual Symposium on Foundations of Computer Science*,pages 422-427, 1989.

[CMS89] B. Chor, M. Merritt, D. Shmoys. Simple Constant Time Consensus Protocols in Realistic Failure Models. *J. ACM,* 36(3):pp. 591-614, 1989.

[H88] M. Herlihy. Impossibility and Universality Results for Wait-Free Synchronization. In *Proceedings 7th Annual ACM Symposium on Principles of Distributed Computing*,pages 276-290, 1988.

[HA88] J. Aspnes, M. Herlihy. Fast Randomized Consensus Using Shared Memory. In *J. Algorithms, Academic Press*, 11,pages 163-183. 1990.

[LA87] M. Loui, H. Abu-Amara. Memory Requirements for Agreement Among Unreliable Asynchronous Processes. In *Advances in Computing Research, JAI Press*, pages 163-183. Volume 4, 1987.

[MS90] C. Martel, R. Subramonian. Asynchronous PRAMs Are (Almost) As Good As Synchronous PRAMs. In *Proceedings, IEEE 30th Annual Symposium on Foundations of Computer Science*,pages 590-599, 1990.

[P89] S. Plotkin. Sticky Bits and Universality of Consensus. In *Proceedings 8th Annual ACM Symposium on Principles of Distributed Computing*,pages 159-176, 1989.

[PSL80] M. Pease, R. Shostak, L. Lamport. Reaching Agreement in the Presence of Faults. *J. ACM,* 27(2):pp. 228-234, 1980.

EFFICIENT CRASH RECOVERY IN SPARSE LOW DIAMETER DISTRIBUTED SYSTEMS*

Tony T-Y. Juang and S. Venkatesan

Computer Science Program, MP 31
University of Texas at Dallas
Richardson, TX 75083-0688
{juang,venky}@utdallas.edu

ABSTRACT

In this paper, a crash recovery algorithm for sparse low diameter networks is presented. Optimistic message logging is used to aid in crash recovery. The resulting algorithm is more efficient (in terms of message complexity) than the other algorithms. Several other issues related to coping with processor failures are also considered and efficient algorithms are presented.

1. INTRODUCTION

In distributed transaction processing systems, there is a need to recover from processor failures quickly to increase the availability of the system. Checkpointing in conjunction with rolling back is a widely used scheme. Processors locally save their history and current states in a stable log from time to time. This process of saving processor states is called checkpointing. If there is a processor failure, the failed processor can restart from the most recently saved state and this is called rolling back. As will be shown subsequently, it may be necessary for some of the non-faulty processors to rollback because of processor failures.

Checkpointing has been widely used and studied by many researchers [1, 3, 5, 6, 11-13, 15, 16]. There are two approaches towards checkpointing and system recovery -- the *synchronous* approach and the *asynchronous* approach. The synchronous approach is to ensure that all processors coordinate their checkpointing such that the global checkpoints are *consistent* [3, 12, 14, 16]. When a failure occurs, processors rollback and restart from their most recent checkpoints. This scheme generates additional messages for each checkpoint and introduce synchronization delays during normal operations.

In the asynchronous approach, each processor takes checkpoints independently and a consistent global state is constructed only when there are processor failures. To aid in minimizing the amount of rolling back at each processor, messages are also logged. Pessimistic and optimistic message logging are the two approaches used.

---

* This work was supported in part by NSF grant CCR-9110177

In pessimistic message logging, each message is synchronously logged to stable storage before it is processed [2, 10]. Since all of the messages received and processed by a processor are readily available in stable log, crash recovery is easy.

In the optimistic message logging approach, messages are logged asynchronously [5, 6, 11, 13]. In this case, the processors log the the received messages in volatile storage. Periodically (or when the processor is idle) each processor saves the contents of its volatile log in stable log and clears the volatile log. Crash recovery is more involved since the processors save their states without coordination. Johnson and Zwaenepoel [5] consider several issues relating to optimistic crash recovery and present algorithms for crash recovery by using optimistic message logging. Sistla and Welch [11] present a recovery algorithm which is a distributed version of the algorithm of Johnson and Zwaenepoel [5] using $O(n^3)$ messages when $O(1)$ information is appended to each message. Juang and Venkatesan [6, 7] improve the message complexity to $O(n^2)$ messages by appending one number and $O(mn)$ messages without adding anything to the application messages.

In this paper, we present a crash recovery technique that is message efficient for low diameter sparse networks. The algorithm appends one number information to each application message and uses $O(mD)$ messages for rolling back where m is the number of links and D is the diameter of the system. This algorithm runs efficiently for networks such as binary hypercube, pyramid, mesh, wheel, etc. For these special cases, the message complexity of this scheme is less than $O(n^2)$ required by the best known scheme of [6].

The paper is organized as follows: In section 2, the computational model and some definitions are presented; in section 3, we present the recovery algorithm which assumes that O(1) information is appended to the application messages; in section 4, several related issues are considered. Finally, section 5 concludes the paper.

2. SYSTEM MODEL

A distributed system can be viewed as a set of n processors which are spatially separated and without shared memory or clock. The processors are connected to each other by a point-to-point communication network that consists of a set of m communication links. The links are assumed to have infinite buffers, to be error-free, and to deliver messages in the order sent. Processors and communication links are said to be asynchronous. Processors directly connected by a communication link are called neighbors.

Several *events* take place in a processor, one event after another and they are denoted by State Transition Intervals (STIs). An event at a processor starts with the receipt of a message from one of its neighbors, and ends with changing its processor state and sending of some messages to some (or all) of its neighbors. We assume that the underlying computation or the application program is *event–driven* where a processor in state s waits until a message m is received, processes the message m, changes its state from s to s', and sends a (possibly empty) set of messages to some of its neighbors. In Figure 2.1, processor p_2

changes its state from s_{22} to s_{23} and send m_3 and m_4 to p_1 and p_3 respectively after it processes message m_2 sent by p_3.

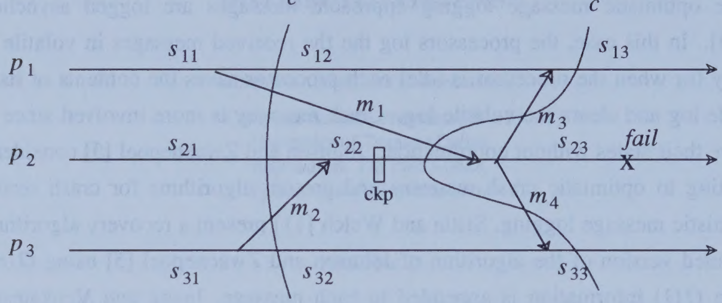


Figure 2.1

Given a processor state s of a processor p, let *event(s)* represent an event e of p such that its processor state immediately following event e is s. For a pair of processors p and q, let $rcd_{p \leftarrow q}(e)$ $(sent_{p \rightarrow q}(e))$ be the sequence of messages received (sent) by p from (to) q up to and including event e of p. State s of processor p is said to be *r_consistent with respect to state t of q* if $rcd_{p \leftarrow q}(event(s))$ is an initial subsequence of $sent_{q \rightarrow p}(event(t))$. Processor states s of processor p and s' of p' are said to be *r_consistent* if state s is r_consistent with respect to s' and state s' is r_consistent with respect to state s. State s_{12} and s_{21} are r_consistent in Figure 2.1. A *global state* of a distributed system is a collection of processor states, one processor state for each processor in the system. A global state is said to be *r_consistent* is every pair of processor states are r_consistent. The global state $\{s_{12}, s_{21}, s_{32}\}$ represented by cut c in Figure 2.1 is r_consistent.

The state of a processor can be lost if the processor fails before saving its state. If the state of a processor p that has sent a message m to q is lost because of the failure of p, then for r_consistency, the state change resulting from the receipt of message m in the processor q must be undone. This requires that the state of processor q must be *rolled back.* We say that the system is in an *optimum consistent state* if the processor states are r_consistent and the amount of rolling back in each processor is minimum.

To recover from processor crashes and restore the system to a consistent state, we use two types of logs - *volatile log* and *stable log* [4, 10]. Accessing volatile logs requires less time, but the contents of a volatile log are lost if the corresponding processor fails. At irregular intervals, each processor (independently) saves the contents of the volatile log in a stable storage and clears the volatile log, and this is called local *checkpointing* . The goal of checkpointing is to eventually log a snapshot of a previous state of the processor in stable storage.

Each time a processor receives a message, it begins a new STI. Each STI is identified by a unique sequential number called *interval index.* Since the resulting state of the receiver depends on the the contents of the message, a *dependency* is created during each STI. For example, in Figure 2.1, state s_{22} depends on state s_{21} and the contents of message m_2. The contents of m_2, in turn, depends on state s_{31} of p_3. Thus, processor state s_{22} depends on processor state s_{31}. This is an example of a direct dependency. Note that this dependency relation is transitive. In the same example, state s_{13} depends on state s_{22} (and the contents of message m_3), and since state s_{22} depends on state s_{31}, processor state s_{13} *transitively depends* on state s_{31}.

Assume that processor p_2 takes a checkpoint when its state is s_{22}. Consider the case when p_2 fails at time t as marked in Figure 2.1. Processor p_2 can restart from state s_{22} since that is the latest processor state of p_2 available. Since the state s_{23} is lost, p_1 and p_3 need to rollback to states s_{12} and s_{32} respectively. A recovery algorithm must find recovery points which are processor states for each processor. For optimality, the recovery points must satisfy the following two conditions:

1. The rolled back processor states should be r_consistent.
2. The amount of rolling back at each processor must be minimum.

In the next two sections, we present recovery techniques that rollback the processor states to a consistent global state from which the application program can resume execution after processor failures.

The efficiency of the recovery algorithms is measured using the worst case message and time complexities. The worst case message complexity counts the maximum number of messages sent by the recovery algorithm among all possible executions. To avoid trivial solutions and to reflect the true cost, the length of a message is bounded by O(log n) bits.

3. ALGORITHM FOR ROLLING BACK

Consider the case when a set of processors fail. Assume that all of the processors have rolled back to the correct recovery points. Let the term 'current recovery point' of a processor denote the STI that the processor has currently rolled back to. The state immediately following this STI is the processor state the processor will start from after recovery if the current recovery point is the final recovery point. It is clear that the final recovery point of any processor cannot transitively depend on an unlogged STI of a failed processor. The recovery algorithm must find, for each processor p, the latest STI i of p such that STI i does not transitively depend on an unlogged STI (of a failed processor) while the immediate successor of i (the STI $i+1$) transitively depends on an unlogged STI (of a failed processor). The recovery problem can now be formulated as follows: for each processor, find its earliest STI that (transitively) depends on an unlogged STI (of a failed processor). The STI immediately preceding that is the correct rollback point.

A sequence $\{e_0, p_{i_0}\}$, m_0, $\{e_1, p_{i_1}\}$, ..., m_{n-1}, $\{e_n, p_{i_n}\}$ is said to be a *dependency chain* if the sequence satisfies the following conditions.

1. Event e_0 is an initial event (the first event) in processor p_{i_0}.
2. For all $j < n$, if $p_{i_{j+1}} = p_{i_j}$ then events e_j and e_{j+1} are two successive events taking place in processor p_{i_j} and message m_j (sent by some processor) was responsible for event e_{j+1} in $p_{i_{j+1}}$.
3. For all $j < n$, if $p_{i_{j+1}} \neq p_{i_j}$, then message m_j was sent by processor p_{i_j} during event e_j of p_{i_j} and this message was received and processed by processor $p_{i_{j+1}}$ during event e_{j+1} of $p_{i_{j+1}}$.

The *length* of a dependency chain is the total number of messages in the chain. In the above sequence, the length of the dependency chain is n (messages m_0, ..., m_{n-1} are in the chain). Note that there may be several dependency chains that end in the same event-processor pair (e, p). For an event e taking place in processor p, the longest dependency chain is a dependency chain $\{e_0, p_{i_0}\}$, ..., m_{s-1}, $\{e_s, p_{i_s}\}$ ending in (e, p) $(e_s = e,\ p_{i_s} = p)$such that s is the largest integer. For each STI i within a processor p, let $LDC_p(i)$ be the length of the longest dependency chain of i. For each STI of a processor, its LDC value can be updated by ensuring that messages generated in an STI contain the LDC of that STI. A message m sent by processor p is stamped with the LDC value of the STI in p during which m is generated. Thus, if m is sent by p during the STI i, then $LDC_p(i)$ is appended to m before m is sent. Let m.LDC be the corresponding LDC value.

Let i be an STI of processor p and let $i+1$ be the STI immediately following i (in processor p) and $PS_p(i)$ be the state of processor p immediately after completing STI i. Let m be the message sent by q that triggered the STI $i+1$ in p. The LDC value of STI $i+1$ is set to a value that is greater than $LDC_p(i)$ and m.LDC. Processor p sets $LDC_p(i+1)$ to $\max\{LDC_p(i)+1, m.\text{LDC}+1\}$. Now, $LDC_p(i+1)$ is appended to each message generated by p in STI $i+1$. In our scheme, we can infer some information about the transitive dependencies of any pair of processors. If STI j of processor q transitively depends on STI i of p, then it is clear that $LDC_p(i) < LDC_q(j)$ if the LDC values are generated correctly. The LDC values are similar to the implementation of logical clocks of [9]. After the end of STI i but before the beginning of STI $i+1$ at p, processor p stores the triple $\{PS_p(i)$[1], $LDC_p(i), m\}$ in its volatile storage where m is the message which triggered STI i.

---

[1] It is not necessary to store the processor state at the end of each STI. It is sufficient to store the processor state only at the beginning of the log.

The recovery procedure consists of two stages -- in the first stage the processors rollback and in the second stage processors resend messages Consider the first stage. With each processor p, we associate two values -- its current recovery point RP(p) and the LDC value (of its current recovery point) LRP(p). It is clear that processor p that has the least value of LRP value (of current recovery point) need not rollback (otherwise there must be another processor with a less value of LRP value as per the rules for assigning LDC values, contradicting the minimality). Thus, processor p decides that its current recovery point is its final recovery point, informs all of its neighbors and remains quiescent for the remainder of the rollback algorithm. The neighbors of p rollback using transitive dependencies and the process continues. This is the main idea of our algorithm. We now present the details.

3.1. Recovery algorithm

It is assumed that no further failures occur when recovery procedure is in progress, each failed processor knows the fact that it had failed when it restarts after failures, and a predetermined processor L will act as a leader or a coordinator. A failed processor sends a *start_recovery* message to processor L which in turn sends it to all of the processors. The recovery procedure at each processor starts on receipt of a *start_recovery* message.

The recovery procedure consists of at most n iterations. During each iteration, at least one processor finds its correct rollback point and remains quiescent. At the end of n iterations, all of the processors rollback to their correct recovery points[2]. Initially, all of the processors are active, and each processor p (including the root L) sets RP(p) to i and LRP(p) to $LDC_q(i)$ where i is the last STI in stable (volatile) log if p is a failed (non-faulty) processor. Then, each failed processor p sends its LRP value to the leader L by sending the message *LRP_value(p*, LRP(p)) and each non_faulty processor q sends message *LRP_value(p ,null*) to L. For each such message, the leader stores the pair (p, LRP(p)) or (q ,*null*) locally in a table called LRP_Table. Processor L starts the first iteration after it receives *LRP_value* messages from each processor in the network. To minimize the total number of messages used, the processors use a breadth first spanning tree rooted at L for message transmission.

During the beginning of each iteration, the leader L locally computes the minimum LRP value (among all of the active processors) using the LRP_Table. Let q be the processor with the minimum LRP value and LRP(q) be the minimum value. Processor L deletes entry (q,LRP(q)) from LRP_Table and sends a *terminate(q)* message to q. In response to a *terminate* message, processor q sends a *rollback(q* , LRP(q)) message to all of its active neighbors and becomes inactive. The *rollback* message is sent so that the active neighbors can rollback based on direct dependencies. Each active neighbor r, on receipt of a *rollback(q* ,LRP(q)) checks if an event that occurred before its own recovery point RP(r)

---

[2] Recovery points are correct if processor states are r_consistent and rollback at each processor is minimum.

directly depends on STI RP(q) of q, and if so, processor r must rollback since processor q has rolled back. On receipt of a *rollback(q,x)* message, processor r scans its (volatile and stable) storage backwards starting from its recovery point RP(r) and finds the triple $\{PS, LDC, m\}$ representing its STI i (if it exists) such that

1. m was received by r from q.
2. m.LDC $> x$
3. i is the smallest STI among those satisfying the above two conditions.

If such an STI i exists, then processor r sets its recovery point RP(r) to STI i-1 and LRP(r) to $LDC_r(i\text{-}1)$. Processor r sends an *LRP_value* (r,LRP(r)) message to L. For each *LRP_value(r, y)* message received by L, processor L replaces the previous entry for processor r by the new entry (r,y) in its LRP_Table. The leader L detects the end of the present iteration when it receives an *LRP_value* message from all of the neighbors of q. The recovery procedure ends when the LRP_Table at L is empty. A formal description of the recovery procedure can be found in [8].

3.2. Correctness

For each failed processor p, let LS_p be the last STI found on the stable log.

Lemma 3.1: At the end of k iterations, the rollback points of the k processors that receive *terminate* messages are r_consistent.

Proof: The proof appears in [8].

Theorem 3.2: At the end of the recovery procedure, all of the processors rollback to the optimum r_consistent recovery points.

Proof: The proof appears in [8].

3.3. Message Complexity

During each iteration, at least one entry is removed from the LRP_Table and hence at least one processor remains inactive. Thus, the total number of iterations is at most n. During the first iteration, all of the processors send their LRP values and this step uses $O(nD)$ messages where D is the diameter of the network. Also, during each iteration, the leader sends a *terminate* message to one processor, the recipient of this message sends a *rollback* message to each active neighbor. These neighbors, in turn update their LRP values and send an *LRP_value* message to the leader. Thus, during each iteration O(D) *rollback* messages and O(d) processors send *LRP_value* messages to L where d is the number of neighbors. Since the worst case minimum number of hops between any two processor is O(D), O(dD) LRP_value messages are sent by each processor. The total number of LRP_value messages is $O(mD)$ as $\Sigma d = 2m$. So, the total number of messages used is $O(mD)$.

Theorem 3.3: The recovery procedure rolls back all of the processors to the optimum r_consistent states using at most $O(mD)$ messages□.

Binary Hypercube:

The binary hypercube having n nodes $n = 2^k$ for some k denoted by $Q(k)$, is the graph whose nodes are labeled with unique binary strings of length k and whose edges connect those binary strings which differ in exactly one position. Clearly, a binary hypercube of n nodes has $n\log n/2$ links and the diameter of the hypercube is $\log n$. Thus, the message complexity of running the recovery algorithm on a binary hypercube is $O(n \log^2 n)$.

Pyramids:

The pyramid of height h, denoted by $P(h)$, is a graph whose nodes are all triples of nonnegative integers (i,x,y), where $0 \le i \le n$ and $1 \le x,y \le 2_i$, and whose edges connect (i,x,y) with the vertices in $\{(i+1,u,v) \mid u \text{ in } \{2x,2x-1\} \text{ and } v \text{ in } \{2y,2y-1\}\}$ as well as with all vertices (i,u,v) such that (x,y) and (u,v) are adjacent nodes in the mesh $[2^i * 2^i]$, for all i $(0 \le i < h)$ and all x,y $(1 \le x,y \le 2^i)$.

$P(n)$ has $1+4+4^2+...+4^h$ nodes. Its diameter is $2*h$ and it has maximum node degree 9. The message complexity on pyramids is $O(9*n*(2h)) = O(n \log n)$.

Meshes:

The d-dimensional mesh of dimensions $a_1, a_2, ..., a_d$, denoted by $[a_1 \times a_2 \times ... \times a_d]$, is the graph whose nodes are all d-tuples of positive integers $(z_1, z_2, ..., z_d)$, where $1 \le z_i \le a_i$, for all i $(0 \le i < d)$, and whose edges connect d-tuples which differ in exactly one coordinate by one. Clearly, $[a_1 \times a_2 \times ... \times a_d]$ has $a_1 \times a_2 \times ... \times a_d$ (represented by n) nodes. Its diameter is $(a_1-1)+(a_2-1)+..+(a_d-1)$ (represented by D) and maximum node degree is $2d$, if each a_i is at least three. The message complexity on meshes is $O(d*n*D)$. For example, in 2-dimension total n nodes square mesh, the message complexity is $O(n\sqrt{n})$.

Wheels:

A wheel is a ring graph with a center node connected to all of the processors in the ring. The diameter is clearly 2 and the message complexity of crash recovery in wheels is $O(2(n-1)*2) = O(n)$.

4. RELATED ISSUES

In the previous section, a recovery algorithm was presented to restore the system to a maximum r_consistent global state in case of processor failures. There are other problems that arise in crash recovery that must be considered.

(i) Resending lost messages -- messages that were generated in STI's that have not been rolled back, and these messages are not in transit nor have they reached their destinations.

(ii) Committing outputs -- determining when a processor can perform certain actions that cannot be undone is important.

(iii) Deleting unwanted information -- deleting obsolete information stored in stable storage is necessary for obvious reasons.

Distributed algorithms for those are presented in [8].

5. CONCLUSIONS

In this paper, the problem of recovering from transient processor failures in distributed systems is considered where the underlying network has low diameter. By appending one number to each application message it is possible to restore the system to the maximum consistent global state and restart the computation. Algorithms were given for rolling back and several related problems. There are several questions that arise out of this work. While the algorithm for rolling back uses $O(mD)$ messages and is more efficient that the algorithm of [6] which uses $O(n^2)$ messages, the question of further improving the message complexity arises. Also, establishing lower bounds on communication complexities of this and related problems, and determining the appropriate times for processors to take local checkpoints so as to minimize the total number of events that are undone at all of the processors are left open.

References

1. Thanawastien, S. , Pamula, R.S., and Varol, Y.L., ''Evaluation of global checkpoint rollback strategies for error recovery in concurrent processing systems,'' *Proceedings of the Sixteenth Annual International Symposium on Fault-Tolerant Computing Systems*, pp. 246-251, 1986.
2. Borg, A., Baumbach, J., and Glazer, S., ''A message system supporting fault tolerance,'' *Proceedings of ACM Symposium on Operating Systems Principles*, pp. 90-99, 1983.
3. Chandy, K.M. and Lamport, L., ''Distributed snapshots: Determining global states of distributed systems,'' *ACM Transactions on Computer Systems*, vol. 3, no. 1, pp. 63-75, 1985.
4. Gray, J., ''Notes on database operating systems: Operating Systems: An advanced course:,'' *Lecture notes in computer science, 60, Springer-Verlag*, pp. 393-481, 1978.
5. Johnson, D. and Zwaenepoel, W., ''Recovery in distributed systems using optimistic message logging and checkpointing,'' *Proceedings of ACM Symposium on Principles of Distributed Computing*, pp. 171-180, 1988.

6. Juang, T-Y. and Venkatesan, S., ''Efficient algorithms for crash recovery in distributed systems,'' *10th Conf. on Foundations of Software Technology and Theoretical Computer Science* , pp. 349-361, 1990.
7. Juang, T-Y. and Venkatesan, S., ''Crash recovery with little overhead,'' *11th International Conference on Distributed Computing Systems*, pp. 454-461, 1991.
8. Juang, T-Y. and Venkatesan, S., ''Efficient Crash recovery in sparse low diameter distributed systems,'' *Technical report UTDCS-5-91, University of Texas at Dallas* , 1991.
9. Lamport, L., ''Time, clocks, and the ordering of events in a distributed system,'' *Communication of the ACM*, vol. 21, no. 7, pp. 558-565, 1978.
10. Powell, M. and Presotto, D., ''Publishing: a reliable broadcast communication mechanism,'' *Proceedings of the ninth ACM Symposium on Operating System Principles*, pp. 100-109, 1983.
11. Sistla, A.P. and Welch, J., ''Efficient distributed recovery using message logging,'' *Proceedings of ACM Symposium on Principles of Distributed Computing*, 1989.
12. Son, S.H. and Agrawala, A.K., ''Distributed checkpointing for globally consistent states of databases,'' *IEEE Transactions on Software Engineering*, vol. 15, no. 10, pp. 1157-1167, 1989.
13. Strom, R.E. and Yemini, S., ''Optimistic recovery in distributed systems,'' *ACM Transactions on Computer Systems*, vol. 3, no. 3, pp. 204-226, 1985.
14. Tamir, Y. and Se´quin, C.H., ''Error recovery in multicomputers using global checkpoints,'' *Proc. 13th IEEE Int. Conf. Parallel Processing*, 1984.
15. Tong, Z., Kain, R., and Tsai, W., ''A Low Overhead Checkpointing and Rollback Recovery Scheme for Distributed Systems,'' *Proceedings of 8th Symposium on Reliable Distributed Systems*, pp. 12-20, 1989.
16. Venkatesh, K., Radhakrishnan, T., and Li, H.F., ''Optimal Checkpointing and Local Recording for Domino-Free Rollback Recovery,'' *Information Processing Letters*, vol. 25, no. 5, pp. 295-304, 1987.

Self-Stabilizing Maxima Finding

Xiaoming Lin and *Sukumar Ghosh*
Department of Computer Science
The University of Iowa
Iowa City, IA 52242

Abstract

Self-stabilization is an abstraction of fault-tolerance for transient failures. A self-stabilizing system has the ability to restore itself to an pre-specified global state from an arbitrary system state. In this paper, we define formally the concept of self-stabilization in terms of fixed points. A self-stabilizing maxima finding algorithm on a ring topology is then presented along with its correctness proof.

1. Introduction

The problem of decentralized maxima finding has many applications in distributed computation. One example is system reconfiguration. A reconfiguration of a system usually involves fault detection, fault isolation, and fault recovery. Typically, a leader is elected among the set of non-faulty processors to coordinate the effort of restoring the system to some operational state. Because the active processor with the largest identification number is normally designated as the leader, the election problem can be considered as a variant of maxima finding problem. Another application of the maxima finding problem can be found in resource allocation in distributed systems. For instance, when a node schedules a job that is too big to run locally, it might be desirable to send the job to the node with the maximum available resources (e.g., CPU time or memory space). In such a case, maximality is a dynamic property of the system.

The past work on the maxima finding problem has focused on achieving either the optimal message complexity [2, 6, 8, 9, 10, 15, 17] or the optimal time complexity [12, 16] under various assumptions. In this paper, we present a *self-stabilizing* maxima finding algorithm on a ring topology. It differs from the previous work in that it tolerates an arbitrary number of transient hardware and communication failures. Informally, a self-stabilizing system refers to a system that can restore itself to an *intended* global state from an arbitrary state in a finite number of steps. A self-stabilizing system has two major advantages over those that are not self-stabilizing: (1) The system will converge to a pre-specified property even in the presence of transient communication and processor failures; (2) No initialization of the system is ever required. The algorithm we propose here is based on the notion of *block*, which is a cluster of locally stable processors. We view a system as composed of blocks, rather than individual processors. Such a hierarchical approach not only simplifies the design of the algorithm, but also permits a straightforward correctness proof.

In [7], Dijkstra introduces the concept of self-stabilization along with three algorithms to demonstrate the feasibility of self-stabilizing systems. Subsequent research on self-stabilization

[3, 4, 5, 11] closely follows Dijkstra's direction in defining mutual exclusion as the goal of self-stabilization. However, the concept of self-stabilization applies to any problem with a well-defined stable property. Examples of such problems are mutual exclusion, maxima finding, and termination detection. In [1], a self-stabilizing algorithm for constructing a spanning tree is presented. The algorithm has the property that, starting from an arbitrary initial state, a fixed point of the system will be reached in a finite number of steps, and the fixed point represents a spanning tree of the underlying network. Since Arora and Gouda's algorithm designates the active processor with the largest identification number as the root of the spanning tree, and every processor in the tree recognizes the identity of the root, it can be considered as a variant of the maxima finding algorithm. The algorithm proposed in this paper also converges to a fixed point, where the active node with the largest identification number is distinguished. However, we have accomplished our goal without the need of constructing a spanning tree.

The remainder of the paper is organized as follows. Section 2 defines the notations and formalizes the concept of self-stabilization in terms of fixed points. Section 3 discusses pitfalls of some existing maxima finding algorithms when communication or processor failures are permitted, and presents a unidirectional self-stabilizing maxima finding algorithm on a ring topology. In section 4, we define the concept of block, and use it to formulate the correctness proof of the algorithm. Section 5 includes a summary of the important features of our algorithm.

2. Self-Stabilization to Fixed Points

We define a system of n processors as a three-tuple, $S = (G, M, \Delta)$, where $G = (V, E)$ is a communication network with the set $V = \{P_0, P_1, ..., P_{n-1}\}$ of processors, and the set E of communication links between processors. M and Δ are state set and transition relations, common to all processors. A global state of the system is described by its configuration. The set of all possible configurations, denoted by Γ, is the Cartesian product of the state sets of all processors in the system. A configuration of the system $\gamma \in \Gamma$ is thus a n-tuple $(a_0, a_1, ..., a_{n-1})$, where $a_i \in M$ for $0 \le i \le n-1$.

A processor P_i with immediate neighbors $P_{v1}, P_{v2}, ..., P_{vk}$ is said to be *enabled* in configuration $\gamma = (a_0, a_1, ..., a_{n-1})$, if and only if there exists some $\delta \in \Delta$, and $\delta(a_i, a_{v1}, a_{v2}, ..., a_{vk}) \neq a_i$. We extend δ to configuration $\gamma = (a_0, a_1, ..., a_{i-1}, a_i, a_{i+1}, ..., a_{n-1})$ to be

$$\delta(\gamma) = \{(a_0, ..., a_{i-1}, b_i, a_{i+1}, ..., a_{n-1}) \mid (\delta(a_i, a_{v1}, a_{v2}, ..., a_{vk}) = b_i) \wedge (b_i \neq a_i), 0 \le i \le n-1\}$$

The system is *enabled* in configuration γ if and only if $\delta(\gamma)$ is not empty. Or equivalently, a system is enabled if there exists some processor P_i, and P_i is enabled in γ. The system is *disabled* in γ, if it is not enabled in γ.

Definition 1. A fixed point *fp* of a system S is a configuration of S such that $\delta(fp)$ is empty. The set of all possible fixed points of the system is denoted by FP.

Informally, a configuration γ is a fixed point of the system if, starting from γ, any further execution of the system would no longer change the state of the system. It should be pointed out that a deadlock state of a system S is also a fixed point of S. However, the reverse is not necessarily true, since we do not require the system to stop execution after reaching a fixed point.

If a system is enabled in γ, and $\gamma' \in \delta(\gamma)$, we write $\gamma \rightarrow \gamma'$ and call $\gamma \rightarrow \gamma'$ an *atomic step* or a *move* of the system. A *computation* of a system is a finite or infinite sequence $\gamma_0, \gamma_1, \ldots$, such that every consecutive pair of configurations in the sequence form an atomic step. A *maximal computation* of the system is either an infinite computation or a finite computation terminated by a fixed point. For ease of notation, we write $\gamma_0 \overset{*}{\rightarrow} \gamma_k$ to denote some finite computation $\gamma_0, \gamma_1, \ldots, \gamma_k$. It is straightforward to see that $\overset{*}{\rightarrow}$ is a transitive relation, i.e., $\gamma_0 \overset{*}{\rightarrow} \gamma_i$ and $\gamma_i \overset{*}{\rightarrow} \gamma_k$ implies that $\gamma_0 \overset{*}{\rightarrow} \gamma_k$. Note that our definition of computation considers only interleaving execution of the system. At each step, exactly one enabled processor is chosen to execute. The serialization of computation can be implemented by using a *central demon* [7], that selects one of the enabled processors to execute at each step. We assume the weak fairness property : if a processor is continuously enabled, it will eventually be chosen by the demon to execute.

With the above notations, we now define the concept of self-stabilization with respect to some stable system property P.

Definition 2. A system $S = (G, M, \Delta)$ is self-stabilizing with respect to system property P, if and only if the following are satisfied:

1. The set of fixed points F_P is not empty.
2. For every fixed point $fp \in F_P$, P holds in fp.
3. From an arbitrary configuration $\gamma \in \Gamma$, every maximal computation converges to some fixed point $fp \in F_P$

In essence, we call a system self-stabilizing, if and only if every maximal computation is finite (and thus contains a fixed point). This is radically different from Dijkstra's definition of self-stabilization, where every maximal computation must be infinite. Our definition is suitable for problems such as maxima finding and termination detection, where the convergence to a certain fixed point is desirable.

3. Maxima Finding in a Ring

Consider an n-processor system $S = (G, M, \Delta)$, where G forms a cycle. The set of processors are labeled $P_0, P_1, \ldots, P_{n-1}$ in counter-clockwise order. We assume each processor P_i has an unique identification number, denoted by *id.i*. For each processor P_i, Processor P_{i-1} is called the predecessor of P_i, and denoted by *Pred.i* . A ring with five processors is shown in Figure 1.

As a motivation for the self-stabilizing maxima finding algorithm, we first analyze the behavior of some existing maxima finding algorithms in the presence of system failures. For most

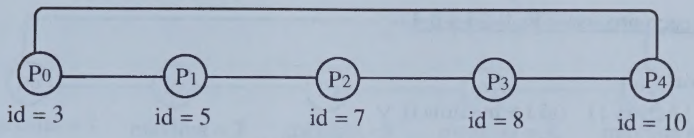


Figure 1. A Ring with Five Processors

maxima finding algorithms, correctness is based on the assumption that the system is free from both processor and communication failures. For instance, Franklin's maxima finding algorithm on ring topology [9] produces erroneous result if, at any stage, the processor with the greatest id becomes inactive. This can happen as a result of either a processor or a communication failure. For Chang and Roberts' algorithm [6], a loss of a single message may cause deadlock if the lost message happens to be the one containing the largest id in the system. Even for algorithms that are designed to tolerate initial system failures [18], it is still essential to assume that all active processors start execution in some predefined initial state.

As in [1], we assume that the number of processors in the ring is bounded by some arbitrary large number N. Each processor P_i contains in its local state a variable *dist.i*, to represent the clockwise distance from P_i to the maxima processor, i.e., the processor with the largest id in the ring. To tolerate transient system failures, it is necessary for each processor to continuously collect information from its neighbors, and update its own state accordingly. For each processor P_i, we maintain a variable *maxima.i*, to denote the id of the maxima processor. Since the initial state of the system can be arbitrary, it follows that some processor P_i can have initial *maxima.i* value that does not match the id of any processor in the ring. This might not cause a problem if *maxima.i* is smaller than the id of some processor in the ring. The complication arises if *maxima.i* is greater than the id of the maxima processor. To filter out such "bad" *maxima.i* value, P_i has to recognize the fact that this choice of maxima value does not match the id of any processor in the system. A simple approach to resolve this is for each processor to collect the id's of all processors in the ring by utilizing a self-stabilizing global snapshot technique [14]. Each processor then decides its own *maxima* value on the basis of its local view of the global system state. However, this approach is not satisfactory for two reasons: (1) Each processor obtains some information that is not relevant to maxima finding, i.e., the id's of the processors other than the maxima processor; and (2) The use of a self-stabilizing global snapshot trivializes our effort of finding an efficient and elegant algorithm.

The self-stabilizing maxima finding algorithm for the n-processor ring is given in Algorithm 1. It consists of three actions, each of them is a guarded assignment with the form

<guard> → <assignment statements>

For each processor P_i, a guard is a boolean expression over its local variables and those of its predecessor. The assignment updates only the variables local to P_i. Note that for some processor, more than one action can be enabled simultaneously at some instance. In such a case, we assume that the demon will arbitrarily choose one of the enabled actions to execute.

```
For each processor $P_i$, $0 \le i \le n-1$

repeat
    {Action 1}  $(id.i > maxima.i) \vee$
                $(id.i = maxima.i \wedge dist.i \neq 0) \vee$
                $(id.i \neq maxima.i \wedge dist.i = 0)$
                    $\rightarrow$ maxima.i:= id.i; dist.i:=0        []

    {Action 2}  $(dist.Pred.i + 1 < N) \wedge (id.i < maxima.Pred.i)$
                    $\rightarrow$ maxima.i:= maxima.Pred.i;
                      dist.i:= dist.Pred.i+1;                    []

    {Action 3}  $(dist.Pred.i + 1 \ge N) \vee$
                $(id.i > id.Pred.i \wedge id.i \ge maxima.Pred.i)$
                    $\rightarrow$ maxima.i:= id.i; dist.i:= 0
forever
```

Algorithm 1. Self-Stabilizing Maxima Finding in a Ring

The first action of Algorithm 1 involves only the variables local to each processor P_i. When P_i detects inconsistency among its local variables, e.g., the value of *maxima.i* is less than the identification number of P_i, or P_i considers itself being the maxima processor but fails to reset *dist.i* to 0, it would simply reset *maxima.i* to *id.i* and *dist.i* to 0. The second action is used when P_i decides to follow the choice of the maxima processor by its predecessor, and updates *dist.i* by adding 1 to the distance value of its predecessor. The last action detects inconsistency between processor P_i and its predecessor, and resets *maxima.i* to *id.i* and *dist.i* to 0.

4. Correctness of Algorithm 1

To show that Algorithm 1 is self-stabilizing, it suffices to show that all three conditions of Definition 2 are satisfied by the algorithm. In the following, we assume variables i, j, k all take values from {0 ... n-1}, where n is the number of processors in the ring and $n \le N$. Throughout this section, we use P_k to denote the maxima processor of the ring. It is straightforward to demonstrate that algorithm 1 has a fixed point *fp* that satisfies

$$P \equiv (maxima.k = id.k \wedge dist.k = 0) \wedge$$
$$(\forall i: i \neq k \ (maxima.i = id.k \wedge dist.i = dist.Pred.i + 1))$$

Informally, in a fixed point state, every processor P_i recognizes P_k as the maxima processor in the ring, and sets *maxima.i* to equal to the identification of P_k, and *dist.i* to equal to the counter-clockwise distance from P_k to P_i.

We now introduce the notion of a block, that is crucial to the reasoning about the correctness of the proposed algorithm.

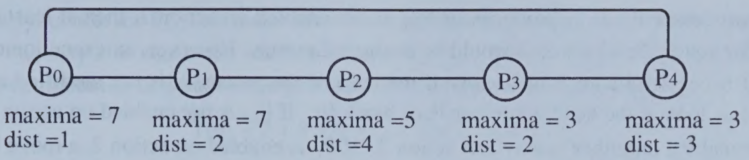


Figure 2. A ring with three blocks: $B1=\{P_0,P_1\}$, $B2=\{P_2\}$, $B3=\{P_3,P_4\}$

Definition 3. A *block* of size *m* is a set of *m* processors $B = \{P_i, P_{i+1}, ..., P_{i+m-1}\}$[1], such that the following two conditions are satisfied for all j, $0 \le j < (m - 1)$:

1. maxima.(i+j) = maxima.(i+j+1)
2. dist.(i+j+1) = dist.(i+j) + 1

Processor P_i and P_{i+m-1} are called the *head* and *tail* of the block, respectively. The maxima value of P_i, that is common to all processors in the block, is called the *M*-value of the block. A ring with three blocks is shown in Figure 2. A block Bi expands when the head processor of the block immediately following Bi[2] makes a move on action 2, and shrinks when the head processor P_i of Bi makes a move, in which case P_{i+1} becomes the new head of the block. The size of a block Bi is reduced to zero if the block consists of a single processor P_i, and P_i makes a move. Depending on the action that it takes, P_i can either start a new block or be merged with another block. For each block $\{P_i, P_{i+1}, ..., P_{i+m-1}\}$, define *MaxSize* to be the difference between the bound on the size of the ring and the distance value of the head processor P_i, i.e., *MaxSize* = (*N* - *dist.i*). The value of *MaxSize* represents the maximum potential expansion of the block. A block with size equal to *MaxSize* can not grow because the distance value of the tail processor of the block equals (*N* - 1), thus preventing the processor immediately following the block to merge into the block by making a move on action 2. The following lemma states an important property of the system.

Lemma 1. If a ring consists of two or more blocks, then the system is enabled.

Proof: Assume that the system has two or more blocks B1, B2, ... Let Bi denote the block that contains the maxima processor P_k. Considering the state of P_k, we have the following two cases:

Case 1: maxima.k = id.k $\wedge$ dist.k = 0. By the definition of block, P_k will be the head of block Bi. Since the system has two or more blocks, the number of processors in block Bi would be less than *n*, the total number of processors in the ring. Let P_j denote the tail processor of block Bi. Then for P_j we have (maxima.j = id.k) $\wedge$ (dist.j < n - 1). Since *n* is bounded by *N*, the above condition implies that the head processor of the block immediately following Bi is enabled on action 2.

Case 2: maxima.k $\neq$ id.k $\vee$ dist.k $\neq$ 0. If maxima.k $\le$ id.k, then the maxima processor P_k is enabled on action 1. So we only need to consider the case where maxima.k > id.k. Note that for any processor P_i, if its predecessor P_{i-1} is not enabled on action 1, then at least one of its guards for action 2 and action 3 would be evaluated to true. However, an execution of action 3 does not necessarily constitute a *move* if the state of the processor is not modified as a result. Now, let us look at the head processor P_i of block Bi. If P_{i-1} is not enabled on action 1, then P_i must be enabled on either action 2 or action 3. If P_i is enabled on action 2, a move by P_i will split P_i from Bi and merge P_i into the block that precedes Bi. If P_i is enabled on action 3, then a move by P_i will also split P_i from block Bi. In such a case P_i starts a new block with *M*-value equal to *id.i*.

---

[1] All indices are taken modulo *n*.
[2] In a counter-clockwise order.

Lemma 2. *fp* is the unique fixed point for Algorithm 1.

Proof: (By contradiction). Assume that the algorithm has another fixed point *fp*' in which $\neg P$ is satisfied. From the definition of *P*, we have

$$\neg P \equiv (\text{maxima.k} \neq \text{id.k} \vee \text{dist.k} \neq 0) \vee$$
$$(\exists \text{i: i} \neq \text{k} \ (\text{maxima.i} \neq \text{id.k} \vee \text{dist.i} \neq \text{dist.Pred.i} + 1))$$

Case 1. $\exists$i: i $\neq$ k (maxima.i $\neq$ id.k $\vee$ dist.i $\neq$ dist.Pred.i + 1)
Under this case, if the ring consists of two or more blocks, then by Lemma 1, the system is enabled. Otherwise all of the processors in the ring form an unique block with *M*-value different from *id.k*. If the *M*-value of the block is less than *id.k*, then P_k will be enabled on action 1. If on the other hand, the *M*-value of the block is greater than id.k, then the head processor P_i of the block would be enabled on either action 2 or action 3. If P_i is enabled on action 2, then a move by P_i will make P_i the new tail of the block. If P_i is enabled on action 3, then P_i will reset *maxima.i* to *id.i*, which is less than the *M*-value of the old block, and P_i starts a new block with *M*-value equal to *id.i*.

Case 2: (maxima.k $\neq$ id.k $\vee$ dist.k $\neq$ 0) $\wedge$ $\neg$($\exists$i: i $\neq$ k(maxima.i $\neq$ id.k $\vee$ dist.i $\neq$ dist.Pred.i + 1))
In this case, it is clear that all processors other than the maxima processor P_k belong to the same block. If P_k does not belong to this block, then the ring will have two blocks and thus the system is enabled by Lemma 1. If P_k does belong to the block that consists of all processors in the ring, then we must have (maxima.k = id.k) $\wedge$ (dist.k $\neq$ 0). But this implies that P_k will be enabled on action 1.

We have shown that in all cases, $\neg P$ implies that the system is enabled, and therefore a state *fp*' in which $\neg P$ is satisfied can not be a fixed point. We conclude that every fixed point of Algorithm 1 satisfies property *P*.

Lemma 3. A computation in which some processor P_i does not move is finite.

Proof: Observe that Algorithm 1 is an unidirectional protocol. A processor makes a move according to its own state and the state of its immediate predecessor. Assume that the processor P_i does not make a move in some computation. Then processor P_{i+1} can make at most two moves, the first move according to action 1, and the second one according to action

2. Similarly, processor P_{i+2} can make two moves before P_{i+1} makes a move. Moreover, each move by P_{i+1} may also cause one additional move by P_{i+2}. Therefore the maximum number of steps in a computation in which some processor P_i does not move is bounded by

$$T(n) = 2 + 4 + 6 + \ldots + 2(n-1) = n(n-1)$$

We now show that starting from an arbitrary global state s, every maximal computation converges to a fixed point state. We demonstrate this by showing that $s \overset{*}{\to} s'$ and $s' \overset{*}{\to} fp$, where s' is some state satisfying the property ($\forall$i: maxima.i $\leq$ id.k).

Lemma 4. From an arbitrary state s, in a finite number of moves the system is guaranteed to reach a state s' in which ($\forall$i: maxima.i $\leq$ id.k) holds.

Proof: Assume that there exists some block B = $\{P_i, P_{i+1}, \ldots, P_{i+j}\}$, whose M-value is greater than the id of the maxima processor P_k. Without loss of generality, we assume that the M-value of B, denoted by M', is the largest among all of the blocks. To simplify the proof, we also assume that B is the only block with M-value M'. By Lemma 3, the head processor P_i of block B will eventually make a move in every computation involving greater than n(n-1) steps. If the size of block B is 1 before P_i makes a move, then the move by P_i will eliminate block B from the ring. Otherwise, a move by P_i can have the following two results: (1) P_i starts a new block with M-value equal to *id.i*; (2) P_i is merged with the block that precedes B, whose M-value is less than M' (by our assumption). In both cases, the net result of a move by P_i is the reduction of the size of B by 1. Since the maximum potential size of a block B is bounded by *MaxSize*, by our earlier analysis, the size of the block B will be reduced to zero in any computation involving *MaxSize**(n(n-1) + 1) steps.

If the ring consists of two or more blocks with M-value equal to M', we can apply the above argument to show that in a finite number of steps, all but one such block will remain. Take two blocks B and B', both having M-value equal to M' and B' being closest to B[3] among all blocks with M-value M'. Lemma 3 asserts that the head of B' should make a move in n(n-1)+1 steps, which would reduce the size of B' by 1 and possibly increase the size of B by 1. Therefore in at most (n-1)(n(n-1) + 1) moves, the size of block B' will be reduced to zero. By an inductive argument, the number of blocks having M-value equal to M' will be reduced to one in a finite number of moves.

In an arbitrary state, the number of blocks with M-value greater than *id.k* is bounded by n, the number of processors in the ring. Algorithm 1 thus guarantees that the system will reach a state where ($\forall$i: maxima.i $\leq$ id.k) holds in a finite number of steps, by first eliminating the blocks with the largest M-value that is greater than *id.k*, and then the blocks with the next largest M-value greater than *id.k*, and so on.

Lemma 5 From a state s' where ($\forall$i: maxima.i $\leq$ id.k) holds, the system converges to the unique fixed point *fp* in a finite number of steps.

---

[3] The distance between two blocks B and B' is defined as the counter-clockwise distance between the heads of B and B'.

Proof: Recall that P_k denotes the maxima processor in the ring. By Lemma 2, P_k will make a move on action 1 if it has not yet set maxima.k = id.k and dist.k = 0. And P_k will remain disabled thereafter. After P_k resets its local state, P_{k+1} will remain enabled on action 2 until it makes a move on that action. When P_{k+1} makes a move on action 2, it will be disabled until P_k makes a move, which we have shown impossible to occur. By the same argument, all processors $P_{k+1}, ..., P_{k+n-1}$ will be disabled in a finite number of steps, and the system converges to the unique fixed point *fp*.

Theorem 1 (Progress of Algorithm 1): From an arbitrary state *s*, $s \overset{*}{\rightarrow} fp$.

Proof: By Lemma 4, Lemma 5, and transitivity of $\overset{*}{\rightarrow}$.

Theorem 2 (Stabilization of Algorithm 1): Algorithm 1 is self-stabilizing with respect to system property *P*.

Proof: By Lemma 2, *fp* is the unique fixed point of Algorithm 1, and property *P* is satisfied in *fp*. By Theorem 1, starting from an arbitrary state s, Algorithm 1 is guaranteed to reach *fp* in a finite number of steps. Therefore all of the three conditions of Definition 2 are satisfied by the algorithm.

5. Conclusion

We have presented a self-stabilizing maxima finding algorithm on a ring topology. The protocol is unidirectional. It tolerates an arbitrary number of transient hardware and communication failures. Together with the self-stabilizing ring orientation presented in [13], our system can also tolerate fail-stop failures when the bound on the number of processors in the ring is known. The main contribution of this paper lies in the hierarchical approach of designing and proving self-stabilizing systems. By exploiting the relationship among the locally stable blocks, we are able to discover a simple and efficient maxima finding algorithm along with a straightforward correctness proof.

Acknowledgements

Acknowledgements are due to Steve Bruell and Hantao Zhang for their helpful comments.

References

[1] Arora, A., and Gouda, M.G., "Distributed Reset," *Proc. of 10th Foundations of Software Tech. & Theoretical Computer Science*, Bangalore, India, Dec., 1990, pp. 316-331.

[2] Bodlaender, Z., and Leeuwen, J. V., "New Upper Bounds for Decentralized Extreme-Finding in a Ring of Processors," *Distributed Algorithms on Graphs*, Gafni, E., and Santoro, N., eds., 1986, Carleton University Press, Ottawa, pp. 27-40.

[3] Brown, G. M., Gouda, M. G., and Wu, C-H., "Token Systems that Self-Stabilize," *IEEE Transactions On Computers*, Vol. 38, No. 6, June 1989, pp. 845-853.

[4] Burns, J. E., "Self-Stabilizing Rings without Demons," *Tech. Report GIT-ICS-87/36*, Georgia Institute of Technology, Nov. 1987.

[5] Burns, J.E., Gouda, M., and Miller, R., "On Relaxing Interleaving Assumptions," *Proc. MCC Workshop on Self-stabilization*, Nov. 1989.

[6] Chang, E., and Roberts, R., "An Improved Algorithm for Decentralized Extreme-Finding in Circular Configurations of Processes," *CACM 22*, 1979, pp. 281-283.

[7] Dijkstra, E. W., "Self-Stabilizing Systems in spite of Distributed Control," *CACM*, Vol. 17, No. 11, 1974, pp. 643-644.

[8] Dolev, M., Klawe, M., and Rodeh, M., "An O(nlogn) Unidirectional Distributed Algorithm for Extreme-Finding in a Circle," *J. Algorithm 3*, 1982, pp. 245-260.

[9] Franklin, W.R., "On an Improved Algorithm for Decentralized Extreme-Finding in Circular Configuration of Processors," *CACM 25*, 1982, pp. 336-337.

[10] Garciaa-Molina, H., "Election in a Distributed Computing System," *IEEE Transactions On Computers*, Vol. c-31, No. 1, January 1982, pp. 48-59.

[11] Ghosh, S., "Self-Stabilizing Distributed Systems with Binary Machines." *Proceedings of the 28th Annual Allerton Conference*, 1990, pp. 988-997.

[12] Gummadi, V., and Abu-Amara, H., "A New Model and Fast Election in Synchronous Network," *Proceedings of the 28th Annual Allerton Conference*, 1990, pp. 858-859.

[13] Kam, M.K., and Bastani, F.B., "A Self-Stabilizing Ring Protocol for Load Balancing in Distributed Real-Time Process Control," *Tech. Rep. #UH-CS-87-8*, Dep. Comp. Sc., Univ. Houston, Houston, Texas, Nov. 1987.

[14] Katz, S., and Perry, K., “Self-Stabilizing Extensions for Message-Passing Systems,” *Proceedings of the MCC Workshop on Self-Stabilizing Systems*, Austin, Texas, 1989.

[15] Pachl, J., Korach, E., and Rotem, D., "A Technique for Improving Lower Bounds for Distributed Maximum-Finding Algorithms," *Proc. 14th ACM Symp. Theory of Computing*, 1982, pp. 378-382.

[16] Peleg, D., "Time-Optimal Leader Election in General Network," *Journal of Parallel Distributed Computing* 8, 1990, pp. 96-99.

[17] Santoro, N., Korach, K., and Rotem, D., "Decentralized Extreme-Finding in Circular Configuration of Processors: and Improved Algorithms," *Congr. Numer 34*, 1982, pp. 401-412.

[18] Taubenfeld, G., "Leader Election in the Presence of n-1 Initial Failures," *Info. Proc. Lett.*, Vol. 33, No. 1, October 1989, pp. 25-28.

Self-Stabilizing Leader Election Algorithms[1]

Mitchell Flatebo Ajoy Kumar Datta
Department of Computer Science
University of Nevada, Las Vegas, NV 89154

1 Introduction

A distributed system consists of a set of loosely connected state machines which do not share a global memory. All the possible global states of the system can be split up into legal and illegal states. A self-stabilizing system is a network of processors, which, when started from an arbitrary (and possibly illegal) initial state, always returns to a legal state in a finite number of steps. In 1974, Dijkstra introduced the property of self-stabilization in distributed systems [2].

This paper will present two self-stabilizing leader election algorithms different from previous algorithms [1][3][4]. After a finite amount of time a leader from among the competing processes will be chosen, and if any error or fault occurs in the system where a new leader needs to be elected, the algorithm will automatically choose a new leader in a finite amount of time. This makes the algorithms more robust and fault tolerant than other leader election algorithms.

2 Leader Election Algorithms

The algorithms presented in this section assume that the n machines are in a ring. The algorithm uses S_i to denote machine i's state. S_{i-1} is the state of machine i's left neighbor $(0 < i \leq n-1)$. S_{i+1} denotes machine i's right neighbor $(0 \leq i < n-1)$. Machine 0's left neighbor is machine $n-1$ and machine $n-1$'s right neighbor is machine 0. Once the algorithm terminates, the states of the machines will be in a consecutive ordering. As an example, after the algorithm terminates, if machine 0 has state 3, the states of all the machines are 3,4,...,$n-1$,0,1,2. The leader will be machine $n-4$ because $S_{n-4} = n-1$ which is the highest state number, and all other machines will know the leader by using $(P_i + (n-1-S_i))$ *mod* n where P_i is the machine's id and S_i is the machine's state. In order to make sure that the consecutive ordering is from 0,1,...,$n-1$, machine 0 forces its state to be in this range if it is initially out of this range by choosing a random integer from 0,1,...,$n-1$ ($random(0, n-1)$).

```
Algorithm 1
{for machine 0}
begin
$(S_0 < 0) \vee (S_0 > n-1) \longrightarrow S_0 := random(0, n-1)$
end
{for other machines}
begin
$S_i \neq (S_{i-1} + 1)\ mod\ n \longrightarrow S_i := (S_{i-1} + 1)\ mod\ n$
end
```

The problem with the previous leader election algorithm is that the unique identifier has to be in 0,1,2,...,$n-1$. This is not always the case because some processes may fail and restart with a different identifier. Other things may also happen to cause identifiers to change. The second algorithm only requires that the state variable of each process, S_i, be in $0 \leq S_i \leq n-1$ while the process id can be anything. This algorithm addresses this problem by introducing an extra variable called $Leader_i$. This variable contains the identifier of the machine that the machine thinks is the leader. Once the system stabilizes, the system will again have a consecutive ordering. Each machine's $Leader_i$ variable will also all be the same, and they will all be the identifier of the machine with state $n-1$.

---

[1] This research was supported in part by the U. S Army Research Office under grant DAAL03-87-G-0004 and by the Information Science Research Institute, University of Nevada, Las Vegas.

Algorithm 2:

```
{for machine 0}
(1) $(S_0 < 0) \vee (S_0 > n-1) \longrightarrow S_0 := random(0, n-1)$
(2) $(Leader_0 \neq Leader_{n-1}) \wedge (S_0 \neq n-1) \longrightarrow Leader_0 := Leader_{n-1}$
(3) $(S_0 = n-1) \wedge (Leader_0 \neq P_0) \longrightarrow Leader_0 := P_0$
{for other machines, $i$}
(4) $S_i \neq (S_{i-1}+1)\ mod\ n \longrightarrow S_i := (S_{i-1}+1)\ mod\ n$
(5) $(Leader_i \neq Leader_{i-1}) \wedge (S_i \neq n-1) \longrightarrow S_i := (S_{i-1}+1)\ mod\ n; Leader := Leader_{i-1}$
(6) $(S_i = n-1) \wedge (Leader \neq P_i) \longrightarrow S_i := (S_{i-1}+1)\ mod\ n; Leader := P_i$
```

3 Conclusions

The self-stabilizing leader election algorithms given in this paper have several advantages. Both algorithms work with either a central or distributed demon. The algorithms are very simple. The first algorithm only had one privilege and one move for each machine. The second algorithm had three privileges and three moves, but this algorithm had many advantages over the first algorithm. The machines could now have arbitrary identification numbers instead of just 0,1,...,$n-1$ as in the first algorithm. The exceptional machine could be any machine (not necessarily machine 0), which is also the case in Algorithm 1. The second algorithm also used an extra variable for the *Leader*. The first algorithm did not need it because the only possible identification numbers were 0,1,...,$n-1$. Another advantage is that the algorithm could be extended easily to work for other topologies, as long as the topology could be mapped into a ring structure.

The advantage of these leader election algorithms is the fault tolerance. After a leader is elected, if a process fails and restarts, the process may have changed its identification number and also its state. If this happens, this change in states creates privileges in the system, and the system once again runs the algorithm until a leader is elected. So, the algorithms automatically elect a new leader if anything happens to put the system into an illegal state. Both algorithms have a worst case complexity of $O(n^2)$. The algorithms do not converge quickly, but the major advantage of these algorithms is the simplicity and fault tolerance.

References

1. Arora A. and Gouda M. "Distributed Reset," *FSTTCS*, 1990.
2. Dijkstra E., "Self-Stabilization in Spite of Distributed Control," *CACM 17*, 1974.
3. Dolev S., Israeli A., and Moran S. "Uniform Dynamic Self-Stabilizing Leader Election," *WDAG*, 1991.
4. Lin X. and Ghosh S. "Self-Stabilizing Maxima Finding," Allerton, 1991.

A Self-Stabilizing Algorithm for Graph Coloring

Sukumar Ghosh [1]
Mehmet Hakan Karaata

Department of Computer Science
University of Iowa
Iowa City, IA 52242

Abstract

This paper describes an algorithm for coloring the nodes of a planar graph with no more than six colors using a self-stabilizing approach. The first part illustrates the coloring algorithm on a directed acyclic version of the given planar graph. The second part describes a self-stabilizing algorithm for generating the directed acyclic version of the planar graph, and combines the two algorithms into one.

Keywords and Phrases: Self-stabilization, distributed algorithm, graph coloring, directed acyclic graph, atomicity.

1 Introduction

The problem of *graph coloring* refers to the assignment of colors (from a given set) to the nodes of a graph, so that no two neighboring nodes have the same color. Of particular interest is the class of planar graphs, that has received substantial attention so far. A proof for 4-colorability already exists [1], and several algorithms [2] [5] [6] for node coloring are available with no more than five colors. The objective of this paper is to introduce a *distributed algorithm* for coloring the nodes of a planar graph with no more than six colors. An interesting feature of the proposed algorithm is that it is *self-stabilizing* [3] [4], so that regardless of the initial colors assigned to the different nodes, each node can eventually adjust its color in such a manner that the

---

[1] This research was supported in part by the National Science Foundation under grant CCR-9109078, and the Old Gold Summer Fellowship from the University of Iowa. The full report has been submitted to a journal for publication.

desired postcondition “no two neighboring nodes have the same color” is satisfied in a finite number of steps.

We consider a planar graph, each node of which is a process. Each nodal process maintains a variable c, that represents the color of that node. The value of the color is locally updated by inspecting the colors of the immediate neighbors only. The program of a typical machine i can be described as $* \; [\; g[1] \rightarrow a[1] \; \square \; g[2] \rightarrow a[2] \; \square$ ... $g[m] \rightarrow a[m] \;]$, where, each *guard* $g[\;]$ is a boolean function of the variables of process i and the variables of its neighboring processes, and each *action* $a[\;]$ updates the variables of process i. Such an action is called a *move*. By definition, the values of c are to be chosen from the set $K = \{0, 1, 2, 3, 4, 5\}$. The problem is to devise an algorithm, so that beginning from arbitrary initial colors assigned to the different nodes, the computation is guaranteed to reach a fixed point, wherein all the guards are false, and the desired postcondition is reached. An advantage of this approach is that such algorithms are inherently robust, since these withstand transient malfunctions of the component machines, as well as dynamic changes in the graph topology.

The basic principle used in this paper is to transform any given planar graph into a directed acyclic graph that satisfies certain specific properties, and execute the self-stabilizing coloring algorithm on the underlying directed graph. In addition to the coloring algorithm, this paper presents a self-stabilizing algorithm for transforming any planar graph into the desired type of *dag*, and suggests a method of combining these into a single algorithm.

The rest of the paper is organized as follows: Section 2 introduces the underlying directed graph model, and presents the self-stabilizing coloring algorithm along with its correctness proof. Section 3 presents a self-stabilizing algorithm for transforming any planar graph into a directed version that satisfies the properties introduced in Section 2, and combines the two algorithms into a single algorithm. Section 4 contains some concluding remarks. Finally, the execution of the proposed algorithm is illustrated in the Appendix.

2 The Principle of Coloring

We first demonstrate the principle of coloring on a directed graph model. Consider a directed graph G_d that satisfies the following two properties:

Property 1 : G_d is acyclic. There is at least one node with indegree = 0, and at least one node with outdegree = 0.

Property 2 : The outdegree of each node in G_d is less than 6.

Each node has a color c, where $c \in K$. The initial values of c are arbitrarily chosen from the set K. Each node is a process that controls the local value of color by monitoring the colors of the neighboring nodes. The problem is to devise an algorithm, so that in a finite number of steps, the computation reaches a fixed-point, and no two neighboring nodes have the same color. The following notations are used to describe the program of each node i in G_d:

- $c[i]$ denotes the color of a node i.
- $succ(i)$ denotes the set of nodes each of which is connected with an outgoing edge from node i.
- $C[i]$ represents the set of colors of all the nodes in $succ(i)$.
- The function $any(S)$ selects any one element from the set S.

Algorithm 1: Self Stabilizing Coloring Algorithm

```
{Program for node $i$}
*[  $\exists$j $\in$ succ(i) | c[i] = c[j] $\longrightarrow$
                  c[i] := any(K-C[i])
]
```

It follows from the above algorithm that, as long as the nodes of the graph are not properly colored, at least one node will have its guard true. Note that nodes with outdegree 0 never change color.

Theorem 1: *Algorithm 1 guarantees a coloring of* G_d.

Sketch of Proof:

Note that $C[i] \subset K$. So, whenever there is a color conflict, a move always exists.

Also, as long as the colors of the nodes in $succ(i)$ remain unchanged, node i will need at most one move to negate the guard.

If the proposition is false, then at least one of the nodes i will make moves infinitely often. This is possible only if at least one of the nodes in $succ(i)$ make moves infinitely often. Since G_d is acyclic, one can extend these arguments, and conclude that a terminal node (with outdegree 0) must also move infinitely often. This is impossible. Hence the proof.

□

In the next section, we describe a self-stabilizing algorithm for transforming a given planar graph G into a directed graph G_d, so that properties 1 and 2 are satisfied.

3 The DAG-generation Algorithm

Consider any planar graph $G = (V, E)$, where V is the set of nodes, and E is the set of edges. With every node i of the graph, we associate an integer variable $x[i]$, that will be used to determine the direction of the edges. The direction of an edge joining a pair of nodes i and j is defined as $i \rightarrow j$ iff $(x[i] < x[j]) \vee ((x[i] = x[j]) \wedge (i < j))$. The edge $i \longrightarrow j$ will be called an *incoming edge* for node j, and an *outgoing edge* for node i.

Each node i of G can start with an arbitrary initial value of $x[i]$. This will lead to unambiguous initial directions to the edges of the graph. Note that regardless of the values of x at the different nodes, the directed version of G *always* remains acyclic. However, property 2 of section 2 may not hold. We need to adjust the values of x at the different nodes in such a manner that, eventually no node has an outdegree greater than five. The following symbols are used in the proposed algorithm:

- $out[i]$ denotes the outdegree of a node i.
- $X[i]$ represents the set of x-values of the nodes in $succ(i)$.

The program of each node i is as follows:

Algorithm 2. Self Stabilizing Dag Generation Algorithm.

```
{Program for node $i$}
*[  out[i] > 5 ⟶
            x[i] :=(max X[i]) + 1
]
```

The crucial issue here is the eventuality of convergence. To prove the correctness of Algorithm 2, we first present a theorem that follows from Euler's polyhedron formula [5]:

Theorem 2: *For any planar graph G with p nodes and e edges,* $e \leq (3p - 6)$.

Corollary 2.1: *In any planar graph, there must exist at least one node whose degree is less than six.*

The above corollary plays an important role in the convergence proof. Every node whose degree is less than six, will be called a *distinguished node.*

Theorem 3: *Algorithm 2 guarantees convergence to the global state in which no node has an outdegree greater than five.*

Sketch of Proof:
Each move by a node i negates the existing guard, and makes *all* its edges *incoming.* This implies that, between two consecutive moves by a node i, *at least six* of its neighboring nodes must make moves.

If the proposition is false, then at least one node j can make moves infinitely often. Thus, at least six neighboring nodes of j must make moves infinitely often. Since the number of nodes is finite, the extension of this argument implies that (i) There exists a subgraph in which every node has a degree of at least six;
(ii) Each node in this subgraph makes moves infinitely often.

However, condition (i) violates Corollary 2.1. Thus no node can make an infinite number of moves, and in a finite number of steps, the algorithm converges to a fixed-pont, where $\forall i$, $out(i) \leq 5$.

□

To combine Algorithms 1 and 2 into a single algorithm, note that Algorithm 1 assumes the existence of no more than five successor nodes for each node in the graph. This may not be true with arbitrary initial values of x. Since we do not assume fairness in choosing the guards, continued execution of the "coloring part" of the algorithm may not converge in a finite number of steps, or the set $\{K - C[i])\}$ may be empty. To overcome this problem, we use an extra guard $out(i) \leq 5$ for the coloring part of the algorithm. The combined algorithm will be as follows:

Algorithm 3. The Combined Algorithm

```
{Program for node i}
*[ ( $out(i) \leq 5$ ) $\wedge$ ( $\exists$ j $\in$ succ(i) | c[i] = c[j]) $\longrightarrow$
                c[i] := any(K-C[i])
[]   ( $out(i) > 5$ ) $\longrightarrow$
                x[i] := (max X[i]) + 1
]
```

4 Conclusion

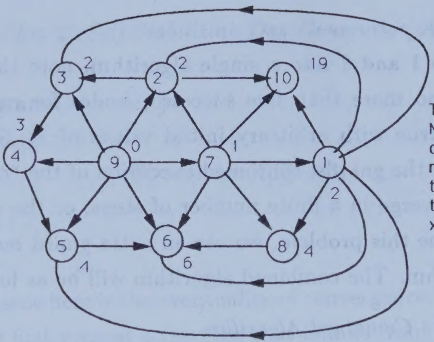
Self-stabilizing distributed algorithms have generated substantial interest in the past few years. However, little work has been done in applying this to the classical problem of graph coloring. This work is a first step in this direction.

The reason behind choosing six colors follows from Corollary 2.1. This algorithm does not guarantee a coloring of the nodes of a planar graph with less than six colors.

Finally, no assumption is made about the magnitude of x, which can be completely arbitrary. Thus, during an indefinite execution, the values of x can indeed become arbitrarily large.

5 Acknowledgement

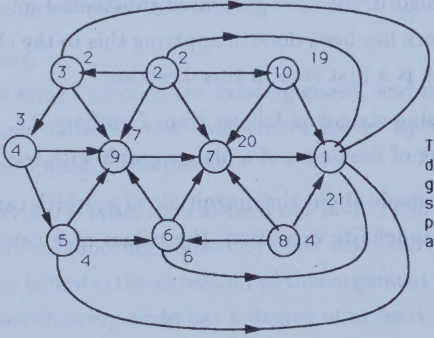
The authors are grateful to Mohamed Gouda, Anish Arora, Jorge Cobb, and Furman Haddix for their valuable comments on an earlier draft of this paper. Acknowledgements are also due to Steve Bruell for helpful criticisms.


3
2
2
2
19
10
3
4
9
0
7
1
1
2
5
6
8
4
6
4

The original graph. Numbers inside circles represent node numbers, and the numbers outside the circles represent x-values

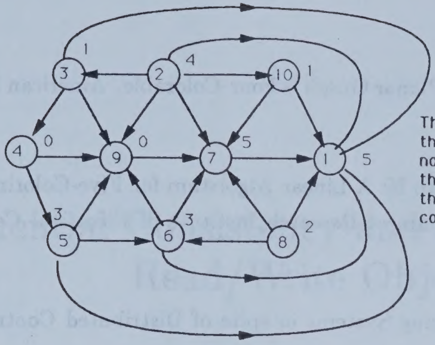
steps	Nodes											
▼	1	2	3	4	5	6	7	8	9	10	Enabled	Move by
initial	2	2	2	3	4	6	1	4	0	19	9, 1	9
step 1	2	2	2	3	4	6	1	4	7	19	7, 1	7
step 2	2	2	2	3	4	6	20	4	7	19	1	1
step 3	21	2	2	3	4	6	20	4	7	19	none	-

Sequence of moves.


3
2
2
2
19
10
3
4
9
7
7
20
1
21
5
4
6
6
8
4

The final directed graph that satisfies properties 1 and 2.

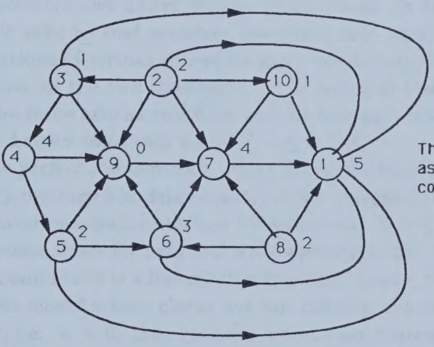
Figure 1: An illustration of the dag-generation algorithm.


1
3
2
4
10
1
4
0
9
0
7
5
1
5
3
5
6
3
8
2

The numbers inside the circles represent node numbers, and the numbers outside the circles represent colors.

steps	Nodes											
▼	1	2	3	4	5	6	7	8	9	10	Enabled	Move by
initial	5	4	1	0	3	3	5	2	0	1	4, 5, 7	5
step 1	5	4	1	0	2	3	5	2	0	1	4, 7	4
step 2	5	4	1	4	2	3	5	2	0	1	7	7
step 3	5	4	1	4	2	3	4	2	0	1	2	2
step 4	5	2	1	4	2	3	4	2	0	1	none	

Sequence of moves.


1
3
2
2
10
1
4
4
9
0
7
4
1
5
5
2
6
3
8
2

The final assignment of colors.

Figure 2: An illustration of the coloring algorithm.

6 References

1. Appel K, Haken F: Every Planar Graph is Four-Colorable. American Mathematical Society 98 (1989).

2. Chiba N, Nishizeki T, Saito N: A Linear Algorithm for Five-Coloring of a Planar Graph. Proc. 17th Symposium of Research Institute of Electrical Communication (1980).

3. Dijkstra EW: Self-stabilizing Systems in spite of Distributed Control. Commun. ACM 17(11): 643-644 (1974).

4. Dijkstra EW: A Belated Proof of Self-stabilization. Distributed Computing 1(1): 5-6 (1986).

5. Harary F: Graph Theory. Addison-Wesley (1980).

6. He Xin: Efficient Parallel and Sequential Algorithms for 4-Coloring Perfect Planar Graphs. Algorithmica (5): 545 - 559 (1990).

Sequential Consistency and Linearizability: Read/Write Objects

Marios Mavronicolas[*] *Dan Roth*[†]
Aiken Computation Laboratory,
Harvard University, Cambridge, MA 02138.

SEPTEMBER 1991

Abstract

We study the cost of implementing read/write objects under two memory consistency conditions, *sequential consistency* and *linearizability*, for shared-memory multiprocessors and under various assumptions on the available timing information. We take as cost measure the worst-case response time in distributed implementations of virtual shared memory consisting of read/write objects and supporting one of the two conditions. It is assumed that processes have clocks that run at the same rate as real time and all message delays are in the range $[d-u, d]$ for some known constants u and d, $0 \leq u \leq d$.

In the *perfect clocks* model, where processes have perfectly synchronized clocks and every message has delay exactly d, we present a family of optimal linearizable implementations, parameterized by a constant β, $0 \leq \beta \leq 1$, for which the worst-case response times for read and write operations are βd and $(1-\beta)d$, respectively.

Our main result is a linearizable implementation, the first known for the *imperfect clocks* model where clocks are not initially synchronized and message delays can vary, i.e., $u > 0$, that achieves worst-case response times of less than $4u + b$ ($b > 0$ is an arbitrarily small constant) and $d + 3u$ for read and write operations, respectively. A simpler version of it achieves identical worst-case response times, but guarantees only sequential consistency. These implementations use synchronization techniques to exploit the lower bound on message delay time and achieve bounds on worst-case response times that depend on the uncertainty u.

---

[*] Supported by ONR contract N00014-91-J-1981. E-mail address: `mavronic@das.harvard.edu`
[†] Supported by NSF grant CCR-89-02500. E-mail address: `danr@das.harvard.edu`

1 Introduction

The *shared-memory* model is an attractive paradigm of an interprocessor communication model, as it provides to the programmers the illusion of a global shared memory across distributed processes. Implementations of shared memory must allow user programs to run "concurrently", i.e., to access shared data by interleaving steps or truly in parallel. However, even in the simplest cases, problems arise since concurrent data accesses cannot be executed instantaneously, and their interleaving raises additional "correctness" problems. Thus, a need arises for a *consistency mechanism* to support the illusion of atomic operations on single copies of memory objects. Such a mechanism must allow operations to be executed concurrently on multiple copies of objects, one at each process, but still guarantee that the operations will appear as if executed atomically, in some sequential order consistent with the order in which individual processes "observe" them to occur. When this order is required to respect also the order of non-overlapping operations at processes, the consistency mechanism is said to guarantee *linearizability* (cf. [3]); otherwise, it is said to guarantee *sequential consistency* (cf. [4]). Clearly, linearizability implies sequential consistency.

Attiya and Welch ([2]) initiated a comparative study of the impact of correctness guarantees provided by sequential consistency and linearizability on the cost of supporting them. They considered caching implementations of read/write objects in non-bused distributed systems and took as cost measure the *worst-case response time* of performing an operation on such objects in the best possible distributed implementation supporting each of the correctness conditions. (Related results also appear in [5, 1, 8].) In this paper, we continue this study by presenting new bounds on the same cost to further enlighten similarities and differences between sequential consistency and linearizability.

As in [2], we consider a collection of application programs running concurrently and communicating through virtual shared memory, consisting of a collection of read/write objects. The shared memory abstraction is implemented by a *memory consistency system* (MCS), which uses local memory at each process node. Each MCS process executes a protocol which defines the actions it takes on operation requests from the application programs. In doing so, the MCS must provide the proper read/write semantics, with respect to the values returned to application programs, throughout the network. At each node, there is a real-time clock, readable by the MCS process at the node, which runs at the same rate as real time. All message delays are in the range $[d-u, d]$, for some known constants u and d, $0 \leq u \leq d$.

For the *perfect clocks* model, where processes have perfectly synchronized clocks and the message delays are constant, i.e., $u = 0$, we present a family of linearizable implementations, parameterized by some constant β, $0 \leq \beta \leq 1$, for which the worst-case response times for read and write operations are βd and $(1-\beta)d$, respectively, where β

may be appropriately chosen. In our algorithms, a read operation returns after time βd, while a write operation returns after time $(1-\beta)d$. By a result of Lipton and Sandberg ([5]) showing a lower bound of d on the sum of the worst-case response times for read and write operations in any linearizable implementation, each of our implementations is optimal. Our implementations naturally generalize those in Theorems 3.2 and 3.3 of [2], which are the special cases of ours where $\beta = 0$ and $\beta = 1$, respectively.

For the more realistic *imperfect clocks* model, where clocks are not initially synchronized and message delays vary, i.e., $u > 0$, we present sequentially consistent and linearizable implementations of read/write objects, the latter being the first known for these objects in this model. Both implementations achieve identical worst-case response times of less than $4u + b$ and $d + 3u$ for read and write operations ($b > 0$ is an arbitrarily small constant), respectively, although the former is simpler and enjoys stronger fairness properties. In our sequentially consistent implementation, a read operation may return a value only after it has resided for time at least u in the local memory; for a write operation, a "time slicing" technique is used, and a process, once in the appropriate time slice, waits for an additional time of d before acknowledging. The linearizable implementation is a modification of the sequentially consistent one, where, making use of an available total order as a tie breaking mechanism, only one write operation among possibly few of them occurring within a short time period becomes effective; in this way, consistent local copies are maintained. Prior to both of these implementations, an initialization phase runs during which processes follow a simple synchronization algorithm to achieve an accuracy of u[1].

Our implementations for the imperfect clocks model make use of the achievable synchronization to exploit the known lower bound on message delivery time and achieve bounds on worst-case response times that depend on the uncertainty u. In the case where the uncertainty u of message delay is sufficiently small, our sequentially consistent implementation significantly improves the one by Attiya and Welch ([2]) achieving worst-case response times of 0 and $6d$ for read and write operations, respectively. Both of our implementations are fairly simple, do not use complicated control mechanisms and are message-economical. Furthermore, our linearizable implementation can be considered as a natural generalization of the perfect clocks one with $\beta = 0$, since, as u tends to 0, it coincides with it and achieves almost identical worst-case response times.

Due to lack of space, the details of some of our definitions, constructions and proofs are omitted here; we refer the reader to [7] for an expanded version of this paper.

---

[1] That is, the maximum difference between local times of any two processes in the system at the same real time is at most u. Although, by the results of Lundelius and Lynch in [6], an accuracy of u is *not* optimal, our synchronization algorithm is extremely simple and an accuracy of u suffices here.

2 Definitions

In this section, we present the formal model and introduce the memory objects, the correctness conditions and the costs of their distributed implementations[2].

A *memory consistency system* (MCS) consists of a collection of n processes $p_1, \ldots, p_n$, each associated with an *application program* P_i; p_i and P_i interact by using *call* and *response* events. (A call event represents a request by an application program to access shared data, while in a response event the corresponding process facilitates such an access.) Processes communicate by sending messages. Each process follows a deterministic local algorithm, invoked by call or message delivery events, that determines for each step the messages to be sent, the response action to be taken and the state transition to be performed.

A *process history* describes what steps the process takes at what real times; it must satisfy certain natural "consistency" conditions. A *timed execution* is a set of histories, one for each process such that: for any processes p and q, there is a one-to-one correspondence between the messages sent by p to q and those received by q from p; the *delay* of any message, defined by the previous correspondence to be the real time of receipt minus the real time of sending, is in the range $[d-u, d]$ for some known constants u and d, $0 \leq u \leq d$; for every process p, at any time at most one call at p is pending, and no two computation steps of p can occur at the same real time. For any timed execution α, let $ops(\alpha)$ be the sequence of call and response events appearing in α in real-time order, breaking ties in case more than one events occur at the same real time by ordering all response events before any call event and then using some available total order.

We consider a collection $\mathcal{X}$ of *read/write objects*, or simply *objects*. Each object X may attain values from a *domain*, a set $\mathcal{V}$ of *values*; a total order $<_{\mathcal{V}}$, known to the processes, is defined on $\mathcal{V}$. Every read/write object X has a *serial specification* (cf. [3]) defining for any $i \in [n]$, $v \in \mathcal{V}$, an ordered pair of events $[\text{Read}_i(X), \text{Return}_i(X, v)]$ which forms a read operation, and an ordered pair $[\text{Write}_i(X, v), \text{Ack}_i(X)]$ which forms a write operation. The set of *admissible operation sequences* for an object X consists of all operation sequences on X that obey the usual read/write semantics; that is, every read operation returns the value of the latest preceding write operation. An operation sequence τ for a collection of processes and objects is *admissible* if, for every object $X \in \mathcal{X}$, the restriction of τ to operations on X is in the set of admissible operation sequences for X, defined by its serial specification.

The definitions of the correctness conditions involve, for a timed execution α, the existence of an operation sequence τ which is a permutation of the operations in α, and possesses certain properties. We say that α is *sequentially consistent* (cf. [4]) if there exists such an admissible τ which also *respects* α: for each process p_i, the restriction of $ops(\alpha)$ to operations of p_i is equal to the restriction of τ to operations of p_i; that is, the sequence of call and response events in α can be permuted to yield a sequence that is admissible and maintains the order of events at each process. We say that α is *linearizable* (cf. [3]) if there exists such an admissible τ which respects α and, in addition, is α-linearizable: whenever the response for operation op_1 precedes the call for operation op_2 in $ops(\alpha)$, then op_1 precedes op_2 in τ; that is, the order of any two non-overlapping operations is preserved. An MCS is a *sequentially consistent implementation* of $\mathcal{X}$ if every timed execution of the MCS is sequentially consistent; similarly, an MCS is a *linearizable implementation* of $\mathcal{X}$ if any timed execution of the MCS is linearizable. The efficiency of an implementation of $\mathcal{X}$ is measured by the worst-case response time for any operation on an object in $\mathcal{X}$. Given a particular MCS and an object X implemented by it, we denote by $|W(X)|$ the maximum time taken by a write operation on X and by $|R(X)|$ the maximum time taken by a read operation on X in any timed execution. Denote by $|W|$ the maximum of $|W(X)|$, and by $|R|$ the maximum of $|R(X)|$, over all $X \in \mathcal{X}$.

---

[2] Our definitions closely follow those in [2].

Finally, we introduce some notation. Fix α to be any timed execution and let $op = [call(op), response(op)]$ be any operation in α. We denote by $t_c(op)$ and $t_r(op)$ the (real) times at which $call(op)$ and $response(op)$ occur in α.

3 Perfect Clocks

In this section, we consider the case where processes have perfectly synchronized clocks and all messages incur a constant and known delay of d. We describe a family of linearizable implementations, RW-Perf-Linear, parameterized by some known to the processes constant β, $0 \leq \beta \leq 1$, which achieves worst-case response times of βd and $(1-\beta)d$ for read and write operations, respectively. Each process p_i keeps a local copy, X_i, of each object X. Upon a $Read_i(X)$ event, p_i waits for time βd; it thereafter reads the value v of its local copy X_i and immediately issues a $Return_i(X, v)$. Upon a $Write_i(X, v)$ event, p_i sends $update(X, v)$ messages to all other processes; it then waits for time $(1-\beta)d$, after which it locally updates X_i to v and issues $Ack_i(X)$. Also, upon receipt of $update(X, v)$ message, p_i immediately updates X_i to v.

Obviously, by the algorithm, the response time for every read operation is βd and the response time for every write operation is $(1-\beta)d$, implying $|R| = \beta d$ and $|W| = (1-\beta)d$. To show that RW-Perf-Linear guarantees linearizability, we fix any timed execution α of RW-Perf-Linear and construct an admissible operation sequence τ, satisfying the linearizability conditions, as follows. We assign a time $t(op)$ to each operation op in α: if op is a read operation, then $t(op) := t_c(op)$, else $t(op) := t_r(op)$). We construct τ as follows: (1) We first order operations in α with respect to t_{op}. (2) For any operations op_1 and op_2 in α such that $t(op_1) = t(op_2)$, we order any of op_1 and op_2 which is a write operation before the other; if op_1 and op_2 are either both reads or both writes, we use $<_V$ to order them. In the full paper, we show that τ is admissible, respects α and is α-linearizable. Thus, we have:

Theorem 3.1 *For each β, $0 \leq \beta \leq 1$,* RW-Imp-Linear *is a linearizable implementation of read/write objects which achieves: $|R| = \beta d$ and $|W| = (1 - \beta)d$.*

4 Imperfect Clocks

In this section, we consider the case where clocks are not initially synchronized and message delays can vary, i.e., $u > 0$. We present sequentially consistent and linearizable implementations, RW-Imp-SeqCons and RW-Imp-Linear, respectively, of read/write objects, which achieve identical worst-case response times of less than $4u + b$ ($b > 0$ is an arbitrarily small constant) and of $d + 3u$ for read and write operations, respectively.

We use a simple synchronization procedure, Synch, which is run during an initialization phase for each of our implementations, to enable the processes achieve a certain amount of accuracy. As soon as a process enters the computation, it broadcasts a special synchronization message *synch* and sets a timer T for time d thereafter. Each process p_i sets its local time to 0 on either the first receipt of some (*synch*) message from any other process or on expiration of T, whichever happens first. In the full paper, we show that Synch synchronizes the system within u, i.e., the maximum difference between the local times of any two processes at any real time after all processes have completed Synch is at most u.

4.1 Sequential Consistency

We start with an informal description of RW-Imp-SeqCons. Each process p_i keeps a local copy, X_i, of each object X and a local variable $timer_i(X)$, a *timer.* Upon a $Read_i(X)$ event, p_i may respond by a $Return_i(X, X_i)$ only if the previous update of X_i occurred at least time u earlier. A "time slicing" technique is used for handling writes: roughly speaking, p_i slices each time period of $3u+b$ into an interval of length $3u$ in which actions on a write request may *not* be initiated, followed by an interval of length b in which they may; $b > 0$ is a "small"constant. Specifically, upon a $Write_i(X, v)$ event and when in the appropriate time interval, p_i broadcasts an $update(X, v)$ message and waits for an additional time d to set X_i to v and issue $Ack_i(X)$. Also, upon receipt of $update(X, v)$, p_i immediately updates X_i to v. Formally, we show:

Theorem 4.1 RW-Imp-SeqCons *is a sequentially consistent implementation of read/write objects which achieves:* $|R| < 4u + b$ *and* $|W| = d + 3u$.

Proof: Obviously, by the algorithm, $|W| = d+3u$. However, the case of reads is not so straightforward and it seems that a deadlock may occur due to successive update events; we show that this is not the case. Fix a timed execution α of RW-Imp-SeqCons and a process p_i such that, for some object X, a $Read_i(X)$ event occurs at time t in α. In the full paper, we show that p_i responds to $Read_i(X)$, by time $t' < t + 4u + b$. Specifically, since the only events that can prevent such a response are $update(X, v)$ events at p_i, we exhibit a family of "quiet" time intervals, $quiet_i(k)$, one for each integer k, with the following properties: p_i receives no update messages in each $quiet_i(k)$; $quiet_i(k)$ has length at least u; two consecutive such intervals, $quiet_i(k)$ and $quiet_i(k+1)$, are separated by an interval, $gap_i(k)$, of length $< 2u+b$. These properties imply that even if $Read_i(X)$ were initiated in some $quiet_i(k)$ interval, but could not be completed in $quiet_i(k)$ because of an update event on X in $gap_i(k)$, it would be completed in $quiet_i(k+1)$ and by time $< t + u + 2u + b + u = t + 4u + b$.

We argue next that RW-Imp-SeqCons is a sequentially consistent implementation. It is possible to explicitly construct an admissible operation sequence τ and show it to respect α, but, instead, we present a different proof method; we construct an order $\leq^*_\alpha$ on the operations in α, which respects the usual read/write semantics for each read/write object X and the ordering of operations at each process. Our construction exploits the following fact shown in the full paper: for any values $v_1, v_2 \in \mathcal{V}$ and $i, i' \in [n]$, $i \neq i'$, if $Return_i(X, v_1)$ precedes $Return_i(X, v_2)$ in $ops(\alpha)$, then $Return_{i'}(X, v_2)$ cannot precede $Return_{i'}(X, v_1)$ in $ops(\alpha)$. That is, for each read/write object X, the order in which values are returned by read operations on X is preserved along different processes and specifies the *value sequence on* X. Based on this sequence, we construct a reflexive order $\leq_\alpha$ on $ops(\alpha)$ which respects the usual read/write semantics for each read/write object and the order of successive operations at the same process, and it is "consistent" with the value sequence on X. Let $\leq^*_\alpha$ be the transitive closure of $\leq_\alpha$. In the full paper, we show that $\leq^*_\alpha$ is a partial order on $ops(\alpha)$ by showing that it is antisymmetric. This implies that any extension of $\leq^*_\alpha$ to a total order yields an admissible sequence of operations which respects α. Since α was chosen arbitrarily, this implies that RW-Imp-SeqCons is a sequentially consistent implementation. ■

4.2 Linearizability

We start with an informal description of RW-Imp-Linear. Each process p_i performs exactly as in RW-Imp-SeqCons, except that, for each read/write object X and in response to $Read_i(X)$, it does not necessarily return the current value x of X_i; rather, it chooses and returns a possibly different one from a set of values of write operations, all occurring within a short period of time. This results not only in that for any read/write object X the order in which values are returned by read operations on X is preserved along different processes, but also in maintaining consistent copies of local memory. As it turns out, this implies linearizability.

We next describe the mechanism by which p_i elects a value to be returned in a read operation; p_i considers only those values that it previously set X_i to, whose local broadcasting time (also broadcasted in the update message) is within $2u$ of that of the message with maximal broadcasting time. (It turns out that the most recent value received must be one of them). To do so, p_i maintains a set $Pend(X)$ of "pending" update messages that it recently (in a precise sense) received. Each time p_i updates X_i to v (either on receipt of $update(X, v)$ or as result of a write operation by p_i), it adds (v, t) to $Pend(X)$, where t is the local time at which $update(X, v)$ was broadcasted in the system. For a $Read_i(X)$ operation, after a value x (broadcasted at local time t) has been the content of X_i for time at least u, p_i considers those elements of $Pend(X)$ whose time component is within $2u$ of the maximal time component in $Pend(X)$. (We show that this is equivalent to considering those elements whose time component is within $2u$ from the current value x, discarding elements that are not in the most recent 2u subinterval); each such element defines a candidate value to be returned; it then returns the maximal (with respect to $<_\mathcal{V}$) of these values. Formally, we show:

Theorem 4.2 RW-Imp-Linear *is a linearizable implementation of read/write objects, which achieves:* $|R| < 4u + b$ *and* $|W| = d + 3u$.

Proof: We observe that RW-Imp-SeqCons and RW-Imp-Linear achieve identical worst-case response times. Thus, we only need to show that RW-Imp-Linear is a linearizable implementation. Fix any timed execution α of RW-Imp-Linear. We construct an admissible sequence of operations τ which respects α and is α-linearizable. In a first phase of our construction, we "serialize" each read and write operation in α to occur at the time of their response in α, breaking ties by ordering all write operations before read ones that occur simultaneously and then using $<_\mathcal{V}$. Clearly, the resulting operation sequence τ' respects α and is α-linearizable; however, it might not be admissible. In a second phase, we trace all admissibility violations in τ', where a read operation on an object X, rop_1, returns not the value written by the latest preceding write operation on X in τ', wop_2, but the value written by some earlier one on X, wop_1. We "fix" each such violation of the admissibility of τ' by permuting wop_2 to immediately precede wop_1 in τ'; in the full paper, we show that there can be no reads on X that return v_2, as written by this wop_2. Let τ be the result of these modifications on τ'. Clearly, τ is admissible, so we only need to show that our modifications preserved the facts that τ' respects α

and is α-linearizable. We do so in the full paper by exploiting the fact that the for each violation case, wop_1 and wop_2 must be completed close enough in time, thus not allowing completion of certain operations between them. ■

5 Discussion and Future Research

We presented sequentially consistent and linearizable implementations of shared memory consisting of read/write objects in perfect and imperfect clock models. In the perfect clocks model, the worst-case response times for read and write operations in an optimal linearizable implementation are βd and $(1-\beta)d$, respectively, for any β, $0 \leq \beta \leq 1$. In the imperfect clocks model, our sequentially consistent implementation achieves worst-case response times of less than $4u + b$ and $d + 3u$ for read and write operations, respectively, where $b > 0$ is an arbitrary small constant. A slight modification of it made use of an available total order to result in a linearizable implementation that achieves identical worst-case response times. Our results, in particular, the upper bound of $d + O(u)$, on the sum of the worst-case response times of reads and writes in *both* sequentially consistent and linearizable implementations, may suggest that sequential consistency and linearizability might be "closer" than thought before. Both of our implementations made heavy use of timing information and, in particular, of the achievable accuracy in the imperfect clocks model to exploit the known lower bound on message delay and achieve bounds on worst-case response times which, unlike previous ones, depend on the message delay uncertainty u.

Our work leaves open several interesting questions. First, it would be interesting to see if our bounds on worst-case response times in the imperfect clocks model can be further improved. Alternatively, it might be possible to improve the lower bounds of $u/4$ and $u/2$ for read and write operations, respectively, shown [2]. Some steps in this direction have already been taken in [8]. We conjecture that a lower bound of $d + u$ on the sum of the worst-case response times of read and write operations holds for any linearizable implementation. Finally, we note that we recently discovered that it is possible to achieve implementations in the imperfect clocks model for which the cost of d is "shared" by the read and write operations, as in the perfect clocks model; our results are reported in [8].

Acknowledgments:

We would like to thank Hagit Attiya for making early versions of [2] and [1] available to us and helpful discussions, and Yishay Mansour for his comments on an earlier version of the paper. We owe special thanks to Harry Lewis for conjecturing Theorem 3.1.

References

[1] H. Attiya, "Implementing FIFO Queues and Stacks," to appear in *5th Workshop on Distributed Algorithms*, Delfi, Greece, 1991.

[2] H. Attiya and J. Welch, "Sequential Consistency versus Linearizability," in *Proceedings of the 3rd ACM Symposium on Parallel Algorithms and Architectures*, pp. 304–315, 1991.

[3] M. Herlihy and J. Wing, "Linearizability: A Correctness Condition for Concurrent Objects," *ACM Transactions on Programming Languages and Systems*, Vol 12, No. 3, July 1990, pp. 463–492.

[4] L. Lamport, "How to Make a Multiprocessor Computer that Correctly Executes Multiprocess Programs," *IEEE Transactions on Computers*, Vol. C-28, No.9 (September 1979), pp. 690–691.

[5] R. Lipton and J. Sandberg, *PRAM: A Scalable Shared Memory*, Technical Report CS-TR-180-88, Princeton University, September 1988.

[6] J. Lundelius and N. Lynch, "An Upper and Lower Bound for Clock Synchronization," *Information and Control*, Vol. 62, Nos. 2/3 (August/September 1984), pp. 190–204.

[7] M. Mavronicolas and D. Roth, "Sequential Consistency and Linearizability: Read/Write Objects," Technical Report, 1991, Aiken Computation Laboratory, Harvard University.

[8] M. Mavronicolas and D. Roth, "On the Costs of Sequential Consistency and Linearizability," in preparation.

A NOVEL DISCRETE EVENT SYSTEM APPROACH TO MODELING AND ANALYSIS OF HYBRID CONTROL SYSTEMS

James A. Stiver and Panos J. Antsaklis
Dept. of Electrical Engineering
University of Notre Dame
Notre Dame, IN 46556
Phone: (219) 239-6916, (219) 239-5792
E-mail: jstiver@bach.helios.nd.edu, flxfsn@irishmvs.cc.nd.edu

1. INTRODUCTION

In this paper, we present a framework for the modeling and analysis of Hybrid Control Systems (HCS). For our purposes, an HCS is a system consisting of a time-driven, continuous-state system, which is being supervised and controlled by a discrete-state symbolic system. In particular, a hybrid control system consists of two rather distinct parts: a system adequately described by a set of difference or differential equations, which shall be called the plant, and a higher level decision making controller, modeled as a discrete-event system. Note that the use of the term "hybrid" is quite distinct from its common use in the control field referring to systems with both digital and analog components. Hybrid control systems are quite common in practice. A familiar hybrid control system is the temperature control of a typical home, if the furnace and air conditioner are modeled as continuous-time systems as they are being controlled by a thermostat which can be regarded as a decision making system. Recently, attempts have been made to study hybrid control systems in a unified, analytical framework [4, 12, 17, 18, 24].

2. AN APPROACH TO MODELING HYBRID CONTROL SYSTEMS

The difficulty in modeling and analysis of HCS arises from their heterogeneous nature. The plant is a continuous-state system which evolves in real time, while the controller is an event driven system which it evolves in logical time. In order for the plant and controller to interact, the HCS must contain an additional part which serves as an interface and allows an exchange of information between the plant and controller. The three parts of a hybrid control system, controller, interface, and plant, can be envisioned as a hierarchical system with the controller as the highest layer, the interface in the middle, and the plant as the lowest layer.

The model described here, and used throughout this work, is a refinement of the above three layer hierarchy made up of four interacting blocks. Two of these blocks, the plant and controller, relate to the corresponding parts of the hierarchy, and the remaining two blocks, the actuator and aggregator, each represent a portion of the interface. The actuator relays commands from the controller to the plant, and the aggregator carries information from the plant to the controller. Figure 2.1 shows the arrangement of these four blocks as well as the names and locations of the signals which provide communication between the blocks. Each of the blocks and its associated signals will be described now.

2.1 Plant

The plant is modeled as a time-invariant, continuous-time system (CTS). This can be used to represent any type of system which evolves in real time and can be suitably approximated by ordinary differential equations. Formally the plant is specified by the following equations:

$$\dot{x}(t) = f(x(t), r(t)) \tag{1}$$
$$z(t) = g(x(t)) \tag{2}$$

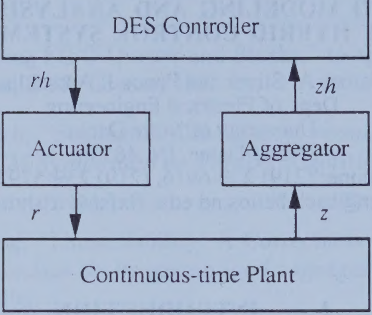


Figure 2.1: Hybrid Control System Model

where $x(t) \in \mathbf{R}^n$ is the plant state, $r(t) \in \mathbf{R}^m$ is the input, and $z(t) \in \mathbf{R}^p$ is the measured output. For the purposes of this work we assume that $z(t) = x(t)$. Note that a discrete-time system (DTS) plant can also be used by replacing equation (1) with $x(k+1) = x(k) + r(k)$.

2.2 Aggregator

The aggregator contains the portion of the interface responsible for relaying information from the plant to the controller. The input to the aggregator is the measured output, z, and the output is zh. The task of the aggregator, therefore, is to convert the continuous-time signal, z, to a discrete-event signal, zh. To accomplish this the aggregator partitions the state space of the plant into a number of regions, where each region is associated with an event, $e_i \in E$, where E is a set of possible events. It is possible that more than one region can be associated with the same event. Whenever the state of the plant moves from one region to another, the aggregator outputs the event associated with the new region. The aggregator then remains silent until the plant state moves into another new region.

Mathematically the aggregator can be specified by a function, α, which maps each point in $\mathbf{R}^n$ to an event in E. α is not an injective function, so it induces equivalence classes in $\mathbf{R}^n$. These equivalence classes form the aforementioned regions in $\mathbf{R}^n$ which partition the state space, and are called *aggregator regions*.

Definition 2.1: The *aggregator regions* generated by the function α in the space $\mathbf{R}^n$ form a partition of $\mathbf{R}^n$ such that two points, $r_1, r_2 \in \mathbf{R}^n$, are in the same aggregator region iff there exists a curve in $\mathbf{R}^n$ from r_1 to r_2 such that $\alpha(r) = \alpha(r_1) = \alpha(r_2)$ for any point, r, on the curve. The set of aggregator regions shall be denoted by **A**. ☐

Thus

$\alpha : \mathbf{R}^n \rightarrow E$	aggregator function
$z(t) = x(t) \in \mathbf{R}^n$	aggregator input
$zh_i \in E$	aggregator output

The action of the aggregator can be written as

$$zh = sp[\alpha(x)] \tag{3}$$

where we abuse notation slightly by allowing α to operate on a signal. *sp* is a function which suppresses repeated events so that the aggregator generates an event only when the plant state first enters a region. The function α is intended to induce equivalence classes which generate reasonably large regions in the state space, large enough such that the plant state can evolve for a non-zero amount of time and remain in the same region. As a result, *zh* will be a discrete and asynchronous signal which has a value only at the times that x moves from one equivalence class of α to another. The model for the aggregator may seem a bit restrictive, but its ability to handle a variety of systems will be discussed later.

2.3 DES Controller

The controller is discrete-event system, meaning it reacts to discrete symbolic input. In this case, the input and output consist of an asynchronous sequence of symbols which are called events. Several modeling strategies have been developed for discrete-event systems. Here the controller is modeled by a deterministic automaton which may have an infinite number of states. The automaton is specified by a quintuple, $\{S, E, C, \delta, \phi\}$, as follows

$S = \{s_1, s_2, s_3, \ldots\}$	set of controller states
$E = \{e_1, e_2, e_3, \ldots\}$	set of events (controller inputs)
$C = \{c_1, c_2, c_3, \ldots\}$	set of commands (controller outputs)
$\delta : S \times E \rightarrow S$	state transition function
$\phi : S \rightarrow C$	output function

As mentioned previously, the automaton may have an infinite number of states, in fact it may have an infinite number of events and commands as well. This model can be used to represent a wide variety of discrete-event systems, it follows the model found in [14] and is similar to those found in [20, 26].

The input signal to the controller is *zh* and the output signal is *rh*. The names reflect the fact that they are the high level signals corresponding to the continuous-time signals *z* and *r*. Whenever an event is received by the controller on *zh*, the controller immediately undergoes a state transition determined by the state transition function,

$$t_i = \delta(t_{i-1}, zh_i) \tag{4}$$

where $t_i \in S$, $zh_i \in E$, and i is a time index. With each state transition, the controller generates an output by the output function,

$$rh_i = \phi(t_i) \tag{5}$$

where $rh_i \in C$. Note that since the state transitions and outputs of the DES controller occur immediately upon receiving an input event, there is no need to include time in the model. The delay between successive state transitions and outputs is determined completely by the input signal, *zh*. Also note that the subscript on a signal, e.g. zh_i, is used to indicate the logical order of the event, state, or command carried on that signal. The subscript on a set member, e.g. e_i, however, refers to a particular member of the set.

2.4 Actuator

In this model the actuator receives a discrete-event signal, *rh*, carrying commands from the controller. The unit in turn produces a continuous-time signal, *r*, which serves as the input to the plant. To produce *r*, the actuator must be programmed with a method to translate each command in C into a numeric value for *r*. The signal, *r*, is piecewise constant for it depends only on the most recent controller command and will not change until the next command is received. In this way the actuator operation is similar to a hold circuit. The action of the actuator is determined by the function γ, where

$\gamma : C \rightarrow \mathbf{R}^m$ actuator function

$r(t) \in \mathbf{R}^m$ actuator output

$rh_i \in C$ actuator input

and described by

$$r = \mathrm{hold}[\gamma(rh)] \tag{6}$$

where we allow γ to operate on a signal, and *hold* is a function which retains the current value until a new argument appears.

2.5 Example: Double Integrator

As an example of an HCS we will consider a system consisting of a double integrator and a simple automaton. The double integrator forms the plant, and it is specified by the equation

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} r. \tag{7}$$

The aggregator partitions the state-space into four regions corresponding to the four quadrants, such that $\alpha(x) = e_i$ if x lies in quadrant i. The controller consists of a three state automata as shown in Figure 2.2 where c_i is the output associated with state s_i. Finally, the actuator maps the three commands, c_1, c_2, and c_3, to -10, 0, and 10 respectively. The behavior of this system is to drive the state of the plant to zero, as can be seen in Figure 2.3 which shows the state trajectory.

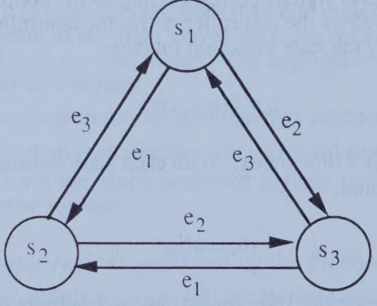


Figure 2.2: DES Controller for Double Integrator

3. A DISCRETE EVENT SYSTEM MODEL FOR HYBRID CONTROL SYSTEMS

The difficulty in analysis presented by Hybrid Control Systems is due to the incompatibility between the models employed in the various blocks which make up the HCS, namely the DES controller and the CTS or DTS plant. A scheme in which a single modeling strategy is used would facilitate analysis. In this section, a method for representing the entire HCS as two interacting DES's will be described. The benefits and limitations of the new model will be described as well as the relationship between the results obtained from the new model and the nature of the original model.

3.1 DES Equivalent Plant

Notice that from the point of view of the DES controller the remainder of the HCS appears as a DES. This is because the input and output of the actuator, plant, and aggregator,

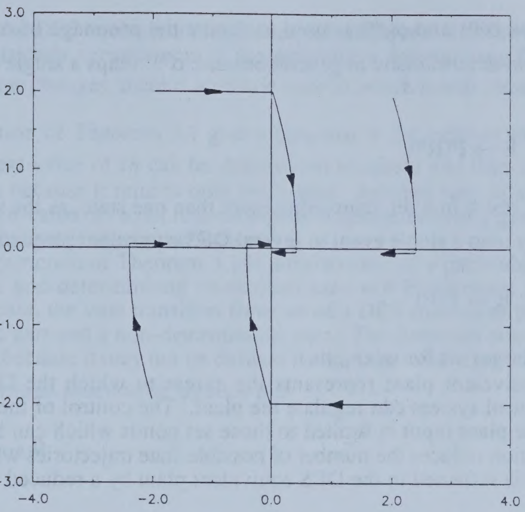


Figure 2.3: Various State Trajectories of the Double Integrator

taken together, are discrete-event signals. This allows the HCS model described in Section 2 to be converted into a discrete-event system model by combining the actuator, plant, and aggregator into a single block. The discrete-event controller remains the same in the new model and the newly formed block constitutes another DES which shall be referred to as the *DES equivalent plant*. Like the controller, the DES equivalent plant is modeled by an automaton, but unlike the controller this automaton may not be deterministic, and the state transitions do not occur instantly when an input is received. However, one can determine exactly when the transitions occur because of the time dependence of the underlying system. The DES equivalent plant is specified by the quintuple $\{P, C, E, \psi, \phi_p\}$ as follows

$P = \{p_1, p_2, p_3, \ldots\}$	the set of states
$C = \{c_1, c_2, c_3, \ldots\}$	the set of inputs (commands)
$E = \{e_1, e_2, e_3, \ldots\}$	the set of outputs (events)
$\psi : P \times C \rightarrow \mathbf{P}(P)$	the state transition function
$\phi_p : P \rightarrow E$	the output function

where **P**(P) denotes the power set of P. Each state in P is uniquely associated with exactly one of these regions, i.e. there is a one-to-one correspondence between the members of the set P and the members of set **A**. This correspondence is such that when the state of the real-time plant is in particular member of **A**, the state of the DES equivalent plant will be the corresponding member of P. The function ϕ_p plays the same role as α because it maps the plant state to an event in E. Notice that the state transition function, ψ, maps to an element of the power set of P, indicating that the automaton is nondeterministic in general. The function ψ is also a partial function in general.

We now express the state transition function, ψ, for the case of a DTS plant. We will use $xh_i \in P$ to denote the state of the DES equivalent plant at time i.

$$\psi(xh_i, rh_i) = \phi_p^{inv}(\alpha(f(\alpha^{inv}(\phi_p(xh_i)), \gamma(rh_i)))) - \{xh_i\} \tag{8}$$

Note that the notation α^{inv} and ϕ_p^{inv} is used to denote the preimage of α and ϕ_p. The DES equivalent plant is non-deterministic in general because α^{inv} maps a single event to a subset of $\mathbf{R}^n$

$$\alpha^{inv} : E \rightarrow \mathbf{P}(\mathbf{R}^n)$$

which will generally result in a set, containing more than one state, as the value of $\psi(xh_i, rh_i)$. In addition, ϕ_p^{inv} may map a single event to several DES equivalent plant states

$$\phi_p^{inv} : E \rightarrow \mathbf{P}(P)$$

resulting in an even larger set for $\psi(xh_i, rh_i)$.

The DES equivalent plant represents the extent to which the DES controller in a particular hybrid control system can regulate the plant. The control of the plant is limited by two factors. First, the plant input is limited to those set points which can be generated by the actuator. This limitation reduces the number of possible state trajectories which can be realized in the plant, and this is reflected in the DES equivalent plant by a reduced number of possible state transitions.

The second limitation on the control of the plant is caused by the aggregator. The aggregator does not provide the DES controller with the exact state of the plant, but rather with a sort of approximation. Thus, based solely on information from the aggregator, the response of the plant to a given input is not known precisely. This effect appears in the DES equivalent plant as nondeterministic state transitions. That is, for a given input and state, the DES equivalent plant may have several possible transitions. The example in Section 3.3 clearly shows this effect. First however, we derive the conditions for determinism.

3.2 Determinism in the DES Equivalent Plant

We would like to know what restrictions can be placed on the hybrid system to ensure that the DES equivalent plant can be represented by a deterministic automaton. This is because the exact behavior of the system can only be determined if the system is deterministic.

Definition 3.1 : A DES equivalent plant is said to be deterministic iff $\forall\ p_i \in P,\ c_j \in C$, we have $\psi(p_i, c_j) \in \{p_l, \text{void}\}$ where $p_l \in P$. □

If ψ is to represent a deterministic automaton, then the expression on the right side of equation (8) must take on values which are sets with only one member.

Theorem 3.1 : The DES equivalent plant will be deterministic iff

$$\forall\ a_1 \in \mathbf{A},\ rh_i \in C\ \exists\ a_2 \in \mathbf{A} \text{ s.t. } x(k+1) \in a_1 \cup a_2$$

where

$$x(k) \in a_1$$

and

$$x(k+1) = f(x(k), \gamma(rh_i)).$$

Proof

Recall that $\mathbf{A}$ is the set of aggregator regions and that there is a one-to-one correspondence between the aggregator regions and the states of the DES equivalent plant. Thus by making the correspondence

$$p_i \Leftrightarrow a_1 \text{ and } p_j \Leftrightarrow a_2,$$

the theorem can be rewritten as

$$\forall\ p_i \in P,\ rh_i \in C\ \exists\ p_j \in P \text{ s.t. } p \in \{p_i, p_j\}$$

where p_i and p are the DES equivalent plant states at time(k) and time(k+1) respectively. Thus Theorem 3.1 is a basically a restatement of the definition a deterministic DES equivalent plant, saying that if the state changes, there is a unique state to which it will change. ☐

The restriction of Theorem 3.1 guarantees that if the current value of *zh* and *rh* are known, then the next value of *zh* can be determined uniquely and thus ψ is a function in the conventional sense because it returns only one value. Another way of stating the theorem is that, under any given value of *rh*, all real-time plant trajectories in any given aggregator region lead to the same subsequent aggregator region.

Often the restriction of Theorem 3.1 is unnecessary. If a particular control objective is to be attained, the non-deterministic transitions may not be relevant to the design of the controller. In any case, the state transition function of a DES equivalent plant can be separated into a deterministic part and a non-deterministic part. The deterministic part will be a partial function in general because it may not be defined for all states and inputs. Let the deterministic part of the state transition function be ψ_d such that

$$\psi_d : P \times C \rightarrow P$$

where

$$\psi_d(xh_i, rh_i) \in \psi(xh_i, rh_i)$$

and ψ_d is defined only for those pairs (p_i, c_j) where $\|\psi(p_i, c_j)\| = 1$. The notation $\|X\|$ refers to the cardinality of set X. Furthermore if the restriction of Theorem 3.1 is met, then

$$\psi(xh_i, rh_i) = \{\psi_d(xh_i, rh_i)\}.$$

The deterministic portion of the DES equivalent plant, described by the function ψ_d, represents in a sense the controllable portion of the DES equivalent plant. The state transitions included in ψ_d can be enabled by the controller and there is no uncertainty in which transition will occur. If these deterministic transitions are sufficient to achieve the control objectives for the system, then the non-deterministic part of the DES equivalent plant is of no consequence.

3.3 Example: Double Integrator

The double integrator system from Section 2.5 is a case of an HCS with a non-deterministic DES equivalent plant. Figure 3.1 shows the state trajectories of the real-time plant under the three available control inputs, along with the aggregator regions. The trajectories which are roughly parallel are under the same input, *r*. As can be seen the aggregator regions this example do not lead to a deterministic DES plant. The reason for this situation is that the aggregator does not preserve enough information about the state of the actual plant. Therefore, by changing the aggregator to add more regions as shown in Figure 3.2, the system can be made to have a deterministic DES equivalent plant. The new aggregator is a refinement of the original which preserves more information about the actual plant state and leads to the deterministic DES equivalent plant shown in Figure 3.3. The original aggregator was sufficient for the stabilizing DES controller of the original example, but the new aggregator will allow any controller to be designed without the worry of unpredictable plant behavior.

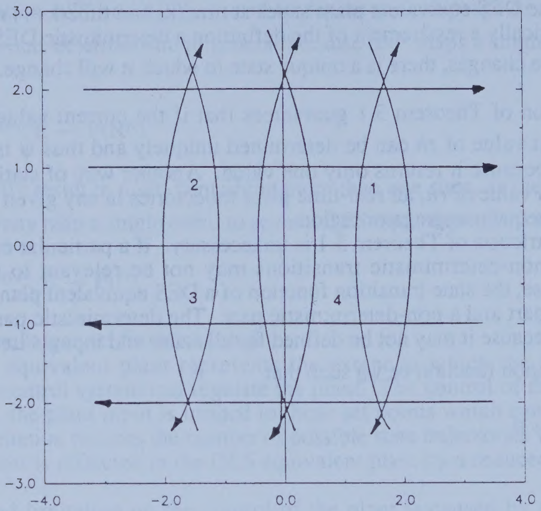


Figure 3.1: Original Aggregator Regions with State Trajectories

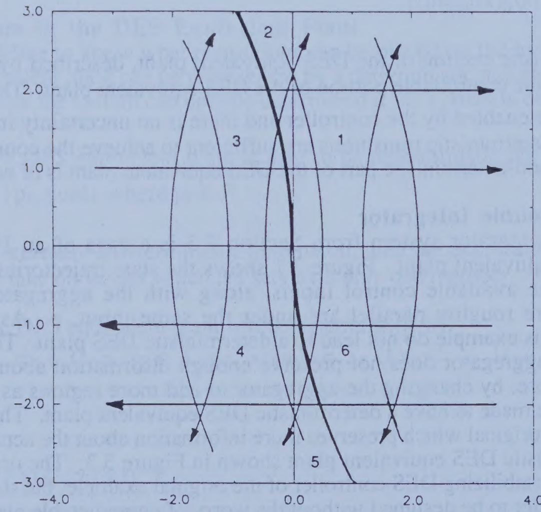


Figure 3.2: New Aggregator Regions with State Trajectories

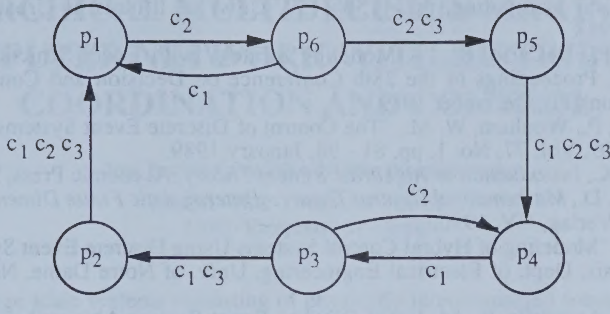


Figure 3.3: The DES Equivalent Plant for the Double Integrator

Final Note

A more thorough treatment of this research can be found in [25], which includes an application of Lyapunov stability to HCS.

7. REFERENCES

[1] Antsaklis, P. J., Passino, K., Wang, S., "Towards Intelligent Autonomous Control Systems: Architecture and Fundamental Issues", *Journal of Intelligent and Robotic Systems*, pp. 315-342, 1989.

[2] Åstrom, J., Wittenmark, B., *Computer-Controlled Systems*, Prentice-Hall, NJ, 1990.

[3] Benveniste, A., Le Guernic, P., "Hybrid Dynamical Systems Theory and Nonlinear Dynamical Systems Over Finite Fields", *New Trends in Nonlinear Control Theory*, pp. 485-495, Springer-Verlag, NY, 1989.

[4] Benveniste, A., Le Guernic, P., "Hybrid Dynamical Systems and the SIGNAL Language", *IEEE Transactions on Automatic Control*, Vol. 35, No. 5, pp. 535-546, May 1990.

[5] Borgne, M., Benveniste, A., Le Guernic, P., "Polynomial Ideal Theory Methods in Discrete Event and Hybrid Dynamical Systems", Proceedings of the 28th Conference on Decision and Control, pp. 2695-2700, Tampa, FL, December 1989.

[6] Cao, X., Ho, Y., "Models of Discrete Event Dynamic Systems", *IEEE Control Systems Magazine*, pp. 69-76, June 1990.

[7] Carroll, J., Long, D., *Theory of Finite Automata*, Prentice-Hall, N.J., 1989.

[8] Cassandras, C., Strickland, S., "Sample Path Properties of Timed Discrete Event Systems", *Proceedings of the IEEE*, Vol. 77, No. 1, pp. 59 - 71, January 1989.

[9] Cassandras, C., Ramadge, P., "Toward a Control Theory for Discrete Event Systems", *IEEE Control Systems Magazine*, pp. 66-68, June 1990.

[10] Fishwick, P., Zeigler, B., "Creating Qualitative and Combined Models with Discrete Events", Proceedings of The 2nd Annual Conference on AI, Simulation and Planning in High Autonomy Systems, pp. 306-315, Cocoa Beach, FL, April 1991.

[11] Francis, B., Wonham, W. M., "The Internal Model Principal of Control Theory", *Automatica*, vol. 12, no. 5, pp. 457-465, May 1976.

[12] Gollu, A., Varaiya, P., "Hybrid Dynamical Systems", Proceedings of the 28th Conference on Decision and Control, pp. 2708-2712, Tampa, FL, December 1989.

[14] Passino, K., "Analysis and Synthesis of Discrete Event Regulator Systems", Ph. D. Dissertation, Dept. of Electrical and Computer Engineering, Univ. of Notre Dame, Notre Dame, IN, April 1989.

[16] Passino, K., Michel, A., Antsaklis, P. J., "Lyapunov Stability of a Class of Discrete Event Systems", Proceedings of the American Control Conference, Boston MA, June 1991. To appear.

[17] Peleties, P., DeCarlo, R., "Modeling of Interacting Continuous Time and Discrete Event Systems : An Example", Proceedings of the 26th Allerton Conference on Comunication,

Control, and Computing, pp. 1150-1159, Univ. of Illinois at Urbana-Champaign, October 1988.

[18] Peleties, P., DeCarlo, R., "A Modeling Strategy with Event Structures for Hybrid Systems", Proceedings of the 28th Conference on Decision and Control, pp.1308 - 1313, Tampa FL, December 1989.

[20] Ramadge, P., Wonham, W. M., "The Control of Discrete Event Systems", *Proceedings of the IEEE*, Vol. 77, No. 1, pp. 81 - 98, January 1989.

[22] Sain, M. K., *Introduction to Algebraic System Theory*, Acedemic Press, NY, 1981.

[23] Sontag, E. D., *Mathematical Control Theory - Deterministic Finite Dimensional Systems*, Springer-Verlag, NY, 1990.

[24] Stiver, J., "Modeling of Hybrid Control Systems Using Discrete Event System Models", M.S. Thesis, Dept. of Electrical Engineering, Univ. of Notre Dame, Notre Dame, IN, May 1991.

[25] Stiver, J., Antsaklis, P., "A Novel Discrete Event System Approach to Modeling and Analysis of Hybrid Control Systems", Control Systems Technical Report #71, Dept. of Electrical Engineering, University of Notre Dame, Notre Dame, IN, June 1991.

[26] Zeigler, B., *Multifacetted Modelling and Discrete Event Simulation*, Acedemic Press, NY, 1984.

[27] Zhong, H.,Wonham, W. M., "On the Consistency of Hierarchical Supervision in Discrete-Event Systems", *IEEE Transactions on Automatic Control,* Vol. 35, No. 10, pp. 1125-1134, October 1990.

HIERARCHICAL MULTIDECISION MAKING ON A COMPUTER NETWORK WITH DISTRIBUTED COORDINATION AND CONTROL

José Tarcísio Costa Filho & Celso Pascoli Bottura
DMCSI - FEE - UNICAMP - CP 6101
13081- Campinas - SP - Brazil

Real large scale systems consisting of physically interconnected subsystems in space, frequently involve several decision makers with different and usually conflicting goals. When more than one decision making is involved, dynamic game theory and hierarchical approaches are useful modelling and studying techniques we may consider [3] [4]. In this sense, the dynamic game, viewed as an organizational task, can be mapped on a computer network where each local decision is made by the associated computer.

In this work, we propose a distributed optimization methodology that involves a temporal and spatial decomposition scheme with coordination distribution based on open-loop Stackelberg and Nash strategies for the solution of the important class of N-person noncooperative dynamic games. This game is characterized by a discrete time linear dynamic model:

$$x(k+1) = Ax(k) + \sum_{i=1}^{N} B_i u_i(k) \quad , \quad x(0) = x_0 \tag{1}$$

where x(k) is a vector of n states variables, $u_i(k)$ (i=1, ..., N) are vectors of m_i control variables, each of which can be manipulated by a different controller, A is n x n matrix and B_i is an n x m_i matrix, and by the associated quadratic objective functions. Each controller must choose an optimal strategy for its own set of control variables over the time period k=0, ...,T-1.

The basic idea of the proposed approach is the multiple coordination of large scale dynamic systems with N controllers, each with a scalar performance index and each controlling a local control variable. Each one of the performance indices is also affected by extra variables (p_i, i=1, ..., N), chosen by the coordination decision makers. Our approach to the problem begins by replacing the lower level problem by an unconstrained parametric optimization subproblems set with the augmented objective functions based on the theory of mathematical programming. For each value of $p_i(k)$, the controls are chosen according to a Nash strategy. The coordination chooses, according to a Nash solution, a value for each $p_i(k)$ such that scalar cost functions at the inferior level are minimized. Thus, the coordination (Nash group) acts as the leader in the Stackelberg sense. For coordination, a gradient type algorithm and a relaxation method is applied. This overall hierarchical structure exhibits two distributed decision levels and decentralized information characteristics, as shown in Fig. 1. The horizontal axes corresponds to the physical decomposition of the global dynamic system, where: a) local controllers (Nash group) at the inferior level are autonomous and can be temporally decomposed; b) the decision makers at the superior level (Nash group) interact with each others. The vertical axes corresponds to the decision process division on the hierarchical structure (Stackelberg strategy) with interactions between levels.

Based on the distributed multilevel decisions structure, we also propose a paralleliza-

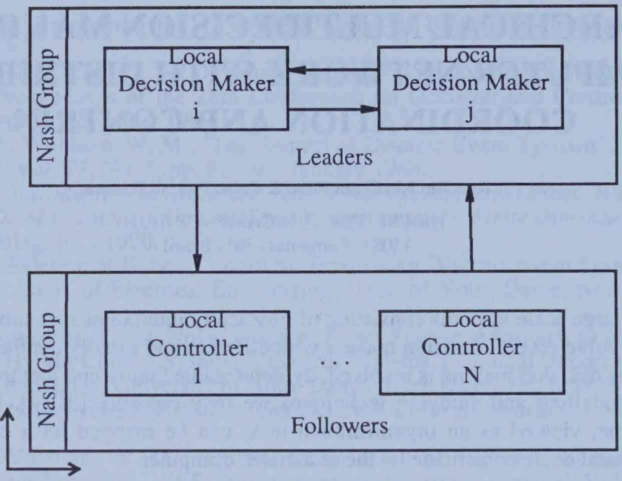


Figure 1- Distributed Hierarchical Decision Process

tion procedure of the above algorithm over a computer network. This procedure consists of: a) for the inferior level: a spatial and functional partition [1] [2] for the distributed and parallel computation, based on the decision makers autonomy resulting from the proposed global system decomposition; b) for the superior level: a spatial partition for the asynchronous parallel computation that results from the coordination distribution and from the decentralized gradient-like algorithm application.

To investigate this proposal with respect to the on-line decision and control process over a network of multiple clusters, using multitransputers and Apollo workstations, we consider the following computational points: i) the amount of required communication for the decision makers and interconnection structure of the computer network; ii) load balancing of the organizational task over the multicomputer architecture; iii) the mapping problem.

References

[1] Costa Filho, J.T. and Bottura, C.P., “On Parallel Hierarchical Multiobjective Optimization of Large Scale Dynamic Systems, Proceedings of the 1991 American Control Conference, Boston, Massachusetts, USA, June 26-28, 1991.

[2] Bottura, C.P. and Costa Filho, J.T., “Distributed Computation Control Using a Conic Like Toolkit and a Multitransputer Network “, Proceedings of the 9th IFAC Workshop on Distributed Computer Control Systems, Tokyo, Japan, September, 1989.

[3] Basar, Tamer and Olsder, Geert Jan, “Dynamic Noncooperative Game Theory, Academic Press, 1982.

[4] Cruz, Jose B., “Leader-Follower Strategies for Multilevel Systems” IEEE Transactions on Automatic Control, Vol. AC-23, No.2, April, 1978.

Optimal Control of Discrete Event Dynamical Systems Using Network Flow Techniques *

Ratnesh Kumar
Department of Electrical Engineering
University of Kentucky
Lexington, KY 40506-0046

Vijay Garg
Department of Electrical and Computer Engineering
University of Texas at Austin
Austin, TX 78712-1084

Abstract

We formalize the notion of optimal control of discrete event dynamical systems (DEDS's) in the framework of Ramadge and Wonham. A DEDS is modeled as a state machine which is controlled by disabling some of its transitions. We define two types of cost functions: a cost associated with disabling a transition in the state machine, and payoff associated with the set of states that remain reachable in the controlled system. The control objective is to design an *optimal* control mechanism (if it exists) so that the *net* cost is minimized. Since a DEDS is represented as a state machine which is a directed graph, network flow techniques are quite naturally applicable in the design of an optimal control mechanism. We also show that the techniques of network flow for optimal control can be used to solve a number of key supervisory control problems of DEDS such as computation of supremal controllable sublanguage, and infimal controllable superlanguage by appropriately defining the cost functions.

Keywords: Discrete Event Dynamical Systems, Supervisory Control, Automata Theory, Optimal Control, Max-flow Min-cut

1 Introduction

Research on supervisory control of discrete event dynamical system (DEDS) was pioneered by Ramadge and Wonham [16]. In [16], a DEDS, also called plant, is modeled as a state machine (SM), and the behavior of a DEDS is described by the language accepted by the corresponding SM. A controller or supervisor disables transitions from the set of *controllable* transitions based on its observations to achieve some desired control objective.

Usually, the control objective is to restrict the plant behavior such that it remains confined within a specific range. Such control objective is studied in [16, 15, 10, 1, 7] etc. There also has been some work in which the control objective is to restrict the plant behavior so that a certain cost function defined along the trajectory of the system is optimized [13, 12, 2, 18].

In [13, 12], a cost function is defined on the set of transitions, and the control objective is to restrict the plant behavior in such a way that after starting at the initial state the plant reaches one of the accepting states along a trajectory of optimal cost. Authors provide an efficient heuristic search based algorithm for solving the problem. In [2] also, a cost function is defined on the set of transitions and the control objective is to restrict the plant behavior so that after starting from *any* state the plant reaches one of the accepting states along a trajectory of optimal cost. In [18] two types of costs, a control cost and a path cost, are defined on the graph representing a plant and the control objective is to determine

---

*supported in part by grants from the Bureau of Engineering Research and University Research Institute, University of Texas at Austin, and NSF Grant CCR-9110605.

the "optimal subgraph" of the original graph which has the maximum total cost. However, only acyclic graphs are considered for analysis.

Our approach to the optimal control problem is conceptually similar to the one considered in [18]. Consider the following setting. We are given the specifications of plant (and its behavior) and that of the desired *target* behavior - a subset of the plant behavior. We associate a cost, called *cost of control*, with any control mechanism that restricts the plant behavior, and a payoff called, *payoff of control*, with the closed-loop behavior thus achieved. Both the cost of control and the payoff of control satisfy some natural properties such as: the cost of no control is zero; the payoff is maximum when the closed-loop behavior equals the target behavior; "more control" costs more; "smaller deviation" of the closed-loop behavior from the target behavior has a larger payoff etc. Conceptually, the cost and payoff functions can be plotted as in the Figure 1 shown below. The control objective is to choose a control mechanism so that the the net cost of control (the difference in cost and payoff of control) is minimized.

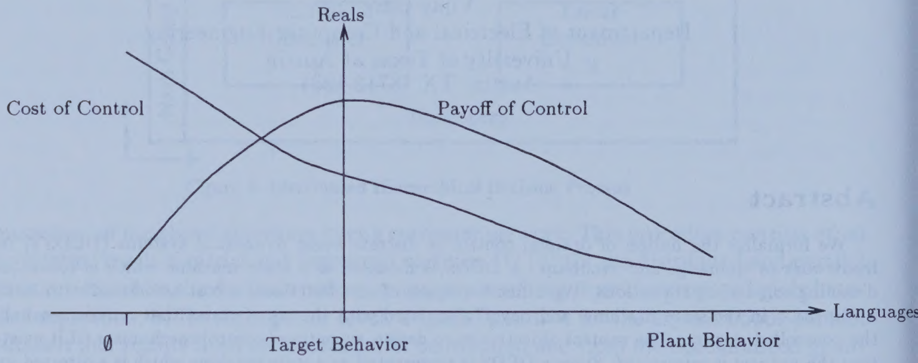


Figure 1: Cost and Payoff of Control

In this paper, we define two types of cost functions. A nonnegative cost function is defined on the set of transitions describing the cost associated with disabling a transition. This cost function corresponds to the cost of exercising any control. A payoff function is defined on the set of states corresponding to the payoff of control. The payoff function may take both negative and positive values depending on whether a state is undesired or desired. Our notion of cost of control and payoff of control is similar to that of "control cost" and "path cost" in [18]. However, the formulation in [18] is restricted to acyclic graphs only. Notice that we do not have a notion of a target behavior in the above framework, but as it will be shown later, we can capture this notion in the above framework quite easily. The problem then is to design a control mechanism that restricts the plant behavior by disabling some transitions in the plant so that the net cost of control is minimized. First, we consider static controllers [5, 8] as in [14, 9] that disable transitions based on the state of the plant. Next we consider the more general case in which dynamic controllers are used.

The motivation of setting an optimal supervisory control problem for DEDS's is twofold: firstly, to introduce a formal framework for supervisory control of DEDS in which control objective is to optimize the difference in cost and payoff of exercising any control; secondly, to present a unified technique for solving a number of key supervisory control problems such as the computation of supremal controllable sublanguage [15, 1, 7], infimal controllable superlanguage [10] etc.

In section 2 we introduce our notation and formally describe the optimal control problem under complete as well as partial state observation. In section 3 we show how the network flow algorithms can be used to solve the problem of optimal control. In section 4 we show that by appropriately defining the cost functions, the problem of optimal control can be specialized to that of computing the supremal controllable and infimal controllable superlanguage. Section 5 summarizes the work presented in this paper and identifies future research.

2 Notation and Problem Formulation

A discrete event dynamical system to be controlled, called plant, is modeled as a state machine [4] denoted as a four tuple $G \stackrel{\text{def}}{=} (X, \Sigma, \delta, x_0)$, where X denotes the set of states of G; Σ denotes the finite set of transitions, also called *events* in context of DEDS's, of G; $\delta : X \times \Sigma \rightarrow X$ denotes the partial state transition function of G; and $x_0 \in X$ denotes the initial state of G. Note that G is a deterministic state machine, i.e. given a state and an event, the next state of G is either undefined (state transition function is a partial function) or uniquely determined by the state transition function. The behavior of G is described by the language $L(G)$ generated by G:

$$L(G) \stackrel{\text{def}}{=} \{s \in \Sigma^\star \mid \delta(x_0, s)!\},$$

where $\Sigma^\star$ denotes the set of finite sequences of events belonging to Σ; the state transition function is extended in an obvious way to the set $X \times \Sigma^\star$; and the symbol "!" is used to denote "is defined".

A supervisor or a controller describes the set of events to be enabled at each state of G based on the record of observed events and states. We define a supervisor, denoted S, to be a map $S : X \rightarrow 2^{\Sigma}$ that describes the set of events $S(x) \subseteq \Sigma$ that are enabled at each state $x \in X$. S is a static supervisor [14, 9, 8] as the control actions are taken based only on the state of G. Readers are referred to [17] for a more general definition of a supervisor. The controlled plant, denoted G_S is another state machine given by the quadruple $G_S \stackrel{\text{def}}{=} (X, \Sigma, \delta_S, x_0)$, where X, Σ, x_0 are as defined above and δ_S denotes the state transition function of the controlled plant:

$$\delta_S(x, \sigma) \stackrel{\text{def}}{=} \begin{cases} \delta(x, \sigma) & \text{if } \sigma \in S(x) \\ \text{undefined} & \text{otherwise} \end{cases}$$

The behavior of the closed-loop system is described by the language $L(G_S)$ generated by the controlled plant. It is clear that $L(G_S) \subseteq L(G)$, as S disables some of the transitions in G.

Remark 2.1 It is shown in [8, 5] that a static supervisor as defined above can be obtained by letting a *subautomaton* [4, 3] of G run in *synchrony* [7] with G.

Next we formally describe the problem of optimal control. Let $c : \Sigma \rightarrow \mathcal{R}^+$ denote a positive cost function defined on Σ, where $\mathcal{R}^+$ denotes the set of positive reals including infinity. The cost $c(\sigma)$ is the cost associated with disabling a transition $\sigma \in \Sigma$. At this point we would like to make a few remarks:

1. The cost function is defined on the set Σ and not on the set $X \times \Sigma$ as is usually the case in control literature. However, this does not result in any loss of generality as the same techniques can be applied to solve the more general control problem.

2. If $c(\sigma) = \infty$ for an event $\sigma \in \Sigma$, then σ should be disabled by no supervisor. Thus the infinite cost of disabling an event captures the notion of *uncontrollable* [16, 17] events.

3. We assume that the cost of control is a one time cost. This can be interpreted in either of the two ways: an event once disabled at some state remains disabled in that state so that on subsequent visits to that state no cost is incurred in disabling that event; or the cost of control is primarily of setting up a control mechanism - a switch for example - and the cost of enabling or disabling the switch is small compared to the one time set up cost.

   The implication of this assumption is that while designing a supervisor we need to consider the cost of exercising control at any state only once irrespective of the number of times that state is visited. Notice that since the control actions depend only on the state of the plant, the same control action is taken each time the same state is visited. Thus it is possible to associate the cost of control to the first time when any state is visited and the control action is taken.

Next, a payoff function $p : X \rightarrow \mathcal{R}$, is defined on the state set of G, where $\mathcal{R}$ denotes the set of reals including the positive and negative infinity. It corresponds to the payoff associated with the states that *remain* reachable in the controlled plant behavior, i.e. the payoff of a state $x \in X$ is $p(x)$ if can be reached in the controlled plant behavior, otherwise the payoff of x equals zero. The payoff function may take a positive or negative value depending on whether the corresponding state is desired or undesired.

1. The notion of desired target behavior can be captured by defining the payoff function to be negative infinity for those states that are undesired, and any fixed positive real for those states that are desired.

The state machine G can be refined [4, 6, 7] with respect to a given target behavior so that the desired and undesired states are uniquely identified and the payoff function is unambiguously defined. This will be explained formally in section 4.

2. We assume (as above for the cost function) that the payoff of control is also a one time payoff, i.e. the payoff of a state that can be reached in the controlled plant behavior does not depend on the number of times that state can be visited. This can be interpreted as follows: any reachable state can either be visited only once (if it is not a node on a cycle) or infinitely many times (if it is a node on a cycle). If a state can be visited only once, then the payoff of control ought to be a one time payoff, and if a state can be visited infinitely many times, then the payoff of control can be associated with the first time when that state is visited.

With the above definitions of the cost and payoff functions we can introduce the optimal control problem.

Definition 2.2 For any supervisor $S : X \rightarrow 2^{\Sigma}$, the *net cost of control*, denoted $C(S)$, is defined to be the difference in cost and payoff of control of using S:

$$C(S) \stackrel{\text{def}}{=} \sum_{x \in Re(G_S)} [\{ \sum_{\sigma \notin S(x)} c(\sigma)\} - p(x)],$$

where $Re(G_S)$ is the set of reachable states[1] in the controlled plant G_S.

$\sum_{\sigma \notin S(x)} c(\sigma)$ is the cost of all the events disabled in state x. Thus $(\sum_{\sigma \notin S(x)} c(\sigma) - p(x))$ is the difference in cost and payoff of control associated with state x. Finally, the difference in cost and payoff of control of using S is obtained by adding the difference in cost and payoff of control associated with each of the states belonging to $Re(G_S) \subseteq X$, the set of reachable states in the controlled plant G_S.

Supervisory Optimal Control Problem 1 (SOCP1): Given a plant $G \stackrel{\text{def}}{=} (X, \Sigma, \delta, x_0)$, a cost function $c : \Sigma \rightarrow \mathcal{R}^+$, and a payoff function $p : X \rightarrow \mathcal{R}$ design a supervisor $S : X \rightarrow 2^{\Sigma}$ such that the difference in cost and payoff of control is minimized, i.e. determine

$$\arg\min_{S} C(S).$$

2.1 Partial State Observation

In SOCP1, a supervisor while deciding its control actions assumes that complete state information of G is available. We pose another optimal control problem in which complete state information is not assumed. To this end we define a mask $M : X \rightarrow Y$ from the state space X to the observation space Y. For each $x \in X$, $M(x) \in Y$ is the state "observed" by a supervisor. A supervisor in this case is given by the map $S : Y \rightarrow 2^{\Sigma}$. Since a supervisor takes a control action based on observing $y \in Y$, given any $y \in Y$ the same control action is taken for all states $M^{-1}(y) \subseteq X$. Thus corresponding to a supervisor $S : Y \rightarrow 2^{\Sigma}$ we can define an equivalent supervisor $S' : X \rightarrow 2^{\Sigma}$ with the constraint C1:

C1: $\forall x_1, x_2 \in X, M(x_1) = M(x_2) \Rightarrow S'(x_1) = S'(x_2)$

Supervisory Optimal Control Problem 2 (SOCP2): Given a plant $G \stackrel{\text{def}}{=} (X, \Sigma, \delta, x_0)$, a cost of control function $c : \Sigma \rightarrow \mathcal{R}^+$, a payoff of control function $p : X \rightarrow \mathcal{R}$, and a mask $M : X \rightarrow Y$ design a supervisor $S : Y \rightarrow 2^{\Sigma}$ (equivalently a supervisor $S' : X \rightarrow 2^{\Sigma}$ satisfying C1) such that the difference in the cost and payoff of control is minimized, i.e. determine

$$\arg \min_{\{S:\ \text{C1 holds}\}} C(S).$$

The difference in SOCP1 and SOCP2 is that in SOCP2 the minimization is taken over the supervisors that also satisfy the constraint C1. However, we will show in the next section that SOCP2 can be reduced to SOCP1 by making a few modifications. In sequel it will turn out that SOCP1 is the basic optimal control problem and many other key supervisory control problems can be reduced to SOCP1 as discussed in section 4.

---

[1] Given a state machine $V \stackrel{\text{def}}{=} (Q, \Sigma, \rho, q_0)$, the set of reachable states $Re(V)$ is recursively defined as: $q_0 \in Re(V)$, and $q \in Re(V) \Rightarrow \rho(q, \cdot) \in Re(V)$.

3 Solution using Network Flow Algorithm

In this section we provide a solution to the two optimal control problems introduced in section 2. The problem is basically to decide for each transition of the state machine G whether to disable it so that the difference in cost of disabling and payoff of the set of states that remain reachable in controlled plant is minimized. In other words, we need to determine an "optimal" partition of the state space X into the set of states that should remain reachable in the controlled plant and the set of remaining states. We determine the "optimal" partition for the control problems posed above by using the max-flow min-cut theorem [11], a powerful tool for partitioning a directed graph.

3.1 Max-Flow Min-Cut Theorem

Interested readers are referred to [11] for a formal and elaborate description of the max-flow min-cut theorem. Informally described, in its simplest form, a *flow network* is represented as a weighted directed graph having a single *source* node from where the flow starts and a single *terminal* node where the flow terminates. The weights on the directed edges of the graph represent the maximum capacities of each of the flow links (the minimum capacity is zero unless specified). Formally,

Definition 3.1 A *flow network* N is a weighted directed graph described by a triple $N = (V, E, u)$, where V denotes the set of vertices or nodes of N; $E \subseteq V^2$ - subset of *ordered* pairs of V^2 - denotes the set of directed edges or links of N; and $u : E \rightarrow \mathcal{R}$ denotes the maximum capacity function of links. V contains two special nodes s and t, the source and the terminal nodes.

The basic flow optimization problem is to determine for a given flow network the maximum possible flow between its source and terminal nodes subject to the capacity constraints.

Definition 3.2 A *cut* of a network N is a partition of V such that s and t are in different partitions. Let $V_s \subseteq V$ and $V_t = V - V_s$ denote the partitions of a cut such that $s \in V_s$ and $t \in V_t$. Then the *capacity* of this cut is defined as:

$$\sum_{(i,j)\in E: i \in V_s, j \in V_t} u((i,j))$$

A *min-cut* is a cut of minimum capacity.

Definition 3.3 A *flow* in a network N is a map $f : E \rightarrow \mathcal{R}^+$ such that

1. $\forall e \in E : f(e) \leq u(e)$.
2. $\forall v \in V, v \neq s, v \neq t : \sum_{v' \in V : (v,v') \in E} f((v, v')) = \sum_{v'' \in V : (v'',v) \in E} f((v'', v))$.

$\sum_{v \in V : (s,v) \in E} f((s, v)) = \sum_{v' \in V : (v',t) \in E} f((v', t))$ is called the *value* of f. A max-flow is a flow of maximum (flow) value.

The max-flow min-cut theorem states the following:

Theorem 3.4 Max-flow Min-cut: [11] Value of max-flow of a flow network equals capacity of min-cut of that network.

Algorithms for computing max-flow can be found in [11]. In this paper, we are interested mainly in computing a min-cut for a flow network, which in view of Theorem 3.4 can be computed using any of the max-flow computations.

3.2 Solution of SOCP1

In the next Theorem we describe a way to solve SOCP1 using max-flow min-cut theorem. A flow network N_G for a state machine G is defined as follows:

Definition 3.5 Given a SM $G \overset{\text{def}}{=} (X, \Sigma, \delta, x_0)$, a flow network, denoted N_G, is defined to be $N_G \overset{\text{def}}{=} (V_G, E_G, u_G)$, where

1. $V_G = X \cup \{s, t\}$.
2. $E_G = \{(x_1, x_2) \in X^2 \mid \exists \sigma \in \Sigma \text{ s.t. } \delta(\sigma, x_1) = x_2\} \cup_{x \in X} \{(x, t)\} \cup \{(s, x_0)\}$.

3. $\forall (x_1, x_2) \in X^2$ s.t. $\exists \sigma \in \Sigma$ with $\delta(x_1, \sigma) = x_2 : u_G((x_1, x_2)) = c(\sigma)$,
$\forall x \in X, u_G((x, t)) = -p(x)$, and
$u_G((s, x_0)) = \infty$.

Remark 3.6 Consider any cut of N_G of finite capacity. Then since the weight of the edge between s and x_0 is infinity, x_0 belongs to the partition containing s. Thus, since all the states are reachable in G (from x_0), any node that belongs to the partition containing s remains reachable (from x_0) after the cut is made. Also, the capacity of a cut equals the difference in sum of costs of all the edges across the cut that are directed from "source side" to "terminal side" and sum of payoffs of all the states that remain reachable (from x_0) after the cut is made.

Theorem 3.7 (Solution of SOCP1) A min-cut of N_G is a solution of SOCP1.

We prove a few Lemmas before proving Theorem 3.7.

Lemma 3.8 If S is an optimal supervisor in SOCP1, then S does not disable a self-loop.

Proof: Let $C(S)$ be the net cost of control of using S. Assume that S disables a self-loop labeled $\sigma^\circ \in \Sigma$ at a state $x^\circ \in X$. Let S' be another supervisor that takes same control actions as S does except it does not disable σ° at x°. Then $C(S') = C(S) - c(\sigma^\circ)$; which contradicts the fact that S is an optimal supervisor, for $c(\cdot)$ is a positive real. □

Remark 3.9 At this point one might suspect that an optimal supervisor for SOCP1 is quite restrictive, as it never disables any self-loops which may be okay for the control problems that we are dealing with but may not be so if one is interested in computing say a supremal controllable sublanguage. However, such a suspicion is only a false one, for if a particular self-loop is not present in a desired target behavior, then it will get omitted when the plant behavior is refined with respect to the target behavior. We make this more explicit in section 4.

Lemma 3.10 If S is an optimal supervisor for SOCP1, then for each $x \in X$, S disables either all transitions leading into x (other than self-loops) or no transition leading into x.

Proof: That S disables no self-loops follows from Lemma 3.8. We must show that for any $x \in X$ if S disables at least one transition leading into x, then it must disable all transitions leading into x. Assume that S disables a transition $\sigma \in \Sigma$ leading into x but does not disable all transitions leading into x. Let $C(S)$ be the net cost of using S. Define another supervisor S' that takes same control actions as S does except it does not disable σ. Then $C(S') = C(S) - c(\sigma)$; which contradicts the fact that S is optimal, for $c(\cdot)$ is a positive real. □

Corollary 3.11 If S is an optimal supervisor for SOCP1, then it either disables no transition or partitions the graph of G.

Proof: Follows from Lemma 3.10. □

The implication of Corollary 3.11 is that an optimal supervisor always induces a cut in the corresponding flow network N_G.

Proposition 3.12 If S is an optimal supervisor for SOCP1, then $C(S)$ equals the capacity of the cut induced by S in N_G.

Proof: Follows from the definition of N_G and $C(S)$ (refer to Remark 3.6). □

The proof of Theorem 3.7 is immediate from Proposition 3.12.

Proof (of Theorem 3.7): It follows from Proposition 3.12 that if S is an optimal supervisor for SOCP1, then $C(S)$ equals the capacity of the cut induced by S in N_G. Hence, from the optimality of S, $C(S)$ must equal min-cut of N_G. □

3.3 Solution of SOCP2

For solving SOCP2 we need to minimize the net cost of control over all supervisors that also satisfy the constraint C1. We show that C1 can be satisfied by making the following modifications in G. The modified state machine will be denoted by $G' = (X', \Sigma', \delta', x_0')$.

Definition 3.13 G' is defined as:

1. $X' = X$.

2. $\Sigma' = \Sigma \cup \{\theta\}; \theta \notin \Sigma$

3. $\delta'(\cdot, \sigma) = \delta(\cdot, \sigma)$ if $\sigma \neq \theta$, and
   $\forall \sigma \in \Sigma, x_1, x_2 \in X$ s.t. $M(x_1) = M(x_2)$:
   $\delta'(\delta(x_1, \sigma), \theta) = \delta(x_2, \sigma)$ and $\delta'(\delta(x_2, \sigma), \theta) = \delta(x_1, \sigma)$.

4. $x'_0 = x_0$.

Thus G' is obtained by adding in G an oppositely directed pair of transitions labeled θ between the pair of states reached by executing a common event from a pair of states that "look alike" (under M).

Next, the cost of control function c is extended to the cost function $c' : \Sigma' \rightarrow \mathcal{R}^+$ with $c'(\sigma) = c(\sigma), \sigma \neq \theta$, and $c'(\theta) = \infty$. Consider for a moment the problem of designing an optimal controller for G' with complete state observation, cost of control given by c', and payoff of control given by p. Since $c'(\theta) = \infty$ no supervisor disables θ in G', i.e. the states connected by an edge labeled θ belong to the same partition of G' under any control (of finite cost). Thus if a pair of states can be reached by executing a common transition say $\sigma \in \Sigma$ from a pair of states that look alike, then σ is either disabled or enabled at both the states that look alike (otherwise the pair of states connected by the edge labeled θ will belong to different partitions). In other words, in G' with cost of control c' the constraint C1 is automatically satisfied.

Thus using the state machine G' and the cost of control function c' SOCP2 can be reduced to SOCP1, which we state without proof.

Theorem 3.14 SOCP2 is equivalent to SOCP1 with G, c replaced by G', c' respectively in SOCP1.

Proof: Follows from the definitions of G', c' and is omitted. □

Definition 3.5 can be used to define a flow network $N_{G'} = (V_{G'}, E_{G'}, u_{G'})$ corresponding to the state machine G'.

Corollary 3.15 A min-cut of N'_G is a solution of SOCP2.

Proof: Follows from Theorem 3.14 and Theorem 3.7. □

4 Applications to Supervisory Control

In view of Theorem 3.14 it is clear that SOCP1 is a basic optimal control problem. We show in this section that a variety of key supervisory control problems of DEDS can be solved in an unified fashion by first reducing these problems to SOCP1 and then using the network flow technique of Theorem 3.7 for arriving at a solution for them. To start with we consider the problem of computing the supremal controllable sublanguage.

4.1 Computing Supremal Controllable Sublanguage

Briefly stated, the problem of computing the supremal controllable sublanguage is [15]: Given a plant $G \stackrel{\text{def}}{=} (X, \Sigma, \delta, x_0)$ with behavior $L(G) \subseteq \Sigma^*$, and a desired target behavior $K \subseteq L(G)$ with K prefix closed [4], compute the supremal sublanguage $K^\uparrow \subseteq K$ such that $K^\uparrow$ is *controllable* [16] with respect to G, i.e. $K^\uparrow \Sigma_u \cap L(G) \subseteq K^\uparrow$, where $\Sigma_u \subseteq \Sigma$ denotes the set of uncontrollable events. A closed form expression for $K^\uparrow$ is given in [1] and an optimal algorithm for computing $K^\uparrow$ is given in [7].

We reduce the problem of computing $K^\uparrow$ to an instance of SOCP1. The basic idea is to refine G with respect to K such that the desired and undesired states can be uniquely identified in G and a payoff function can be unambiguously defined. Once G is refined an appropriate cost and payoff function is defined so that the solution of SOCP1 yields the generator for $K^\uparrow$.

First we refine G with respect to K. Let $V \stackrel{\text{def}}{=} (Q, \Sigma, \rho, q_0)$ be a trim deterministic state machine that generates K. The graph of V is completed by adding a dump state $d \notin Q$ to it, i.e. the completion of V is another state machine $V' \stackrel{\text{def}}{=} (Q', \Sigma, \rho', q_0)$ with $Q' \stackrel{\text{def}}{=} Q \cup \{d\}$, and

$$\rho'(q, \sigma) \stackrel{\text{def}}{=} \begin{cases} \rho(q, \sigma) & \text{if } \rho(q, \sigma)! \\ d & \text{otherwise} \end{cases}$$

It is clear that $L(V') = \Sigma^\star$, for all the transitions are defined in all the states of V'. Consider the synchronous composition [5, 7] of state machines G and V': $G\Box V' \overset{\text{def}}{=} (X \times Q', \Sigma, \alpha, (x_0, q_0))$, where

$$\alpha((x, q'), \sigma) \overset{\text{def}}{=} \begin{cases} (\delta(x, \sigma), \rho'(q', \sigma)) & \text{if } \delta(x, \sigma)! \text{ and } \rho'(q', \sigma)! \\ \text{undefined} & \text{otherwise} \end{cases}$$

It is shown in [7, 5] that $L(G\Box V') = L(G) \cap L(V')$. Since $L(V') = \Sigma^\star$, it follow that $L(G\Box V') = L(G)$. Since $K \subseteq L(G)$, every string in K is accepted by $G\Box V'$. $G\Box V'$ is called refinement of G with respect to K. Notice that the first coordinate of a state in $G\Box V'$ belongs to the state of G and the second coordinate to the state of V'. It is readily shown that a state with second coordinate d is reached by executing those and only those strings that are in $L(G) - K$. A state in $G\Box V'$ with second coordinate d is thus an "undesired" state, as it does not correspond to the target behavior, and all the other states are "desired" states, as they correspond to the target behavior K. Hence, a payoff function $p : X \times Q' \to \mathcal{R}$ is defined as:

$$p((x, q')) \overset{\text{def}}{=} \begin{cases} -\infty & \text{if } q' = d \\ p_0 & \text{otherwise} \end{cases},$$

where $p_0 \in \mathcal{R}^+$ is any positive real. Since the payoff of a state in $G\Box V'$ with second coordinate d is negative infinity it must remain unreachable in any controlled plant (otherwise the net cost of control will be infinity). Thus with the above definition of payoff function the undesired states can be avoided in the controlled plant behavior. Since the payoff of all other states is p_0, a positive real number, as many such states as possible must remain reachable in a controlled plant for the payoff to be as large as possible. Thus with the above definition of payoff function the maximality of the reachable desired states can also be ensured.

Next we define a cost function. Since, some of the events in G are uncontrollable and cannot be disabled by any supervisor, we define the cost function $c : \Sigma \to \mathcal{R}^+$ as:

$$c(\sigma) \overset{\text{def}}{=} \begin{cases} \infty & \text{if } \sigma \in \Sigma_u \\ 0 & \text{otherwise} \end{cases}$$

Thus with the above definition of cost function it is ensured that no uncontrollable event is disabled by any supervisor (otherwise net cost of control will be infinity), whereas no cost is incurred in disabling other events.

With the above definitions of cost and payoff of control functions, we can pose a SOCP1 for the state machine $G\Box V'$. Our next Theorem tells that the computation of $K^\uparrow$ is achieved by solving SOCP1 for $G\Box V'$.

Theorem 4.1 Computation of $K^\uparrow$ reduces to solving SOCP1 for $G\Box V'$.

Proof: Let S be an optimal solution for SOCP1 for $G\Box V'$. Then from Corollary 3.11 it follows that S partitions G into the set of states that remain reachable in controlled plant and all the remaining states.

First we show that the controlled plant behavior is a sublanguage of K. Since $p((\cdot, d)) = -\infty$, a state with second coordinate d is not reachable in controlled plant (otherwise net cost of control is infinite). Thus controlled plant behavior is a sublanguage of K.

Next we show that the controlled plant behavior is a controllable sublanguage of K. This follows from the facts that none of the uncontrollable events are disabled (otherwise an infinite cost is incurred), i.e. supervisor S is a *complete*[2] supervisor, and that the controlled plant behavior is controllable if and only if S is a complete supervisor [7].

Finally, we show that the controlled plant behavior is the supremal controllable sublanguage of K. Assume for contradiction it is not true. Consider the supervisor S' that corresponds to the partition of supremal controllable sublanguage. That such a partition does exist is shown in [7]. Then there exists at least one state with second coordinate unequal to d that is reachable in $(G\Box V')_{S'}$ and unreachable in $(G\Box V')_S$. Thus $C(S') \leq C(S) - p_0$, which shows that S is not optimal. □

We summarize the computation of $K^\uparrow$ using network flow techniques:

Algorithm 4.2 Let G denote the plant and V the generator for the target behavior K.

1. Determine V', SM obtained by completing the graph of V.

2. Determine $G\Box V'$.

---

[2] A complete supervisor is one that disables no uncontrollable events [7, 5].

3. Define cost function $c : \Sigma \to \mathcal{R}^+$ and payoff function $p : X \times Q' \to \mathcal{R}$ as described above.
4. Determine the flow network $N_{G\Box V'}$ for SM $G\Box V'$ using Definition 3.5.
5. Determine min-cut of $N_{G\Box V'}$ to obtain the generator for $K^{\uparrow}$.

4.2 Computing Infimal Controllable Superlanguage

Next we consider the problem of computing infimal controllable superlanguage [10], which briefly stated is: Given a plant G and desired target behavior K compute $K^{\downarrow} \supseteq K$ the infimal superlanguage of K such that $K^{\downarrow}$ is controllable.

We use the same notations as described in subsection 4.1. The following modification is made to payoff function.

$$p((x, q')) \overset{\text{def}}{=} \begin{cases} -p_0 & \text{if } q' = d \\ 0 & \text{otherwise} \end{cases}$$

Since payoff of visiting a state with second coordinate d is defined to be a negative real as few such states are visited as possible in the controlled plant. Thus with the above definition of the payoff function the minimality of the reachable states that do not correspond to the target behavior is ensured. The following modification to the network $N_{G\Box V'}$ is made.

1. An edge of weight infinity is added between the source node s and each of the states in $G\Box V'$ with second coordinate not equal to d.

This modification in $N_{G\Box V'}$ ensures that all the states with second coordinate not equal to d, i.e. the states corresponding to the desired target behavior K, remain reachable (from x_0) in the controlled plant. The above two modifications together result in a partition corresponding to the generator for $K^{\downarrow}$. This we state in next Theorem without proof.

Theorem 4.3 Algorithm 4.2 (with modified payoff function and network $N_{G\Box V'}$) can be used for computing $K^{\downarrow}$.

Proof: Same as the proof of Theorem 4.1 and is omitted. □

5 Conclusion

We have formally introduced the problem of optimal supervisory control for DEDS by introducing the notions of cost and payoff of using a controller. Cost of control is incurred when an event is disabled by a controller and payoff of control is incurred for the states that remain reachable in the controlled plant. The control objective is to optimize the net cost of control. This is formulated as SOCP1 for the case of complete observation and SOCP2 for the case of incomplete state observation. We show that a solution to SOCP1 can be obtained as a min-cut for an associated flow network, and a solution for SOCP2 is obtained by reducing an instance of SOCP2 to an instance of SOCP1. Thus SOCP1 is concluded to be a basic optimal supervisory control problem for DEDS. As an application of the theory and framework developed in this paper we show that many of other key supervisory control problems such as computation of supremal controllable sublanguage, computation of infimal controllable sublanguage can be reduced to instances of SOCP1. Thus above theory can serve as a unified computational framework for many supervisory control problems.

Research on identifying more applications of the above theory is in progress and we hope to provide computation techniques using network flow theory for supervisory control under partial observation, designing stabilizing supervisors etc. The goal is to present an unified computational framework for various types of supervisory control problems of DEDS and if possible relax some of the assumptions made in the theory so as to broaden its scope.

References

[1] R. D. Brandt, V. K. Garg, R. Kumar, F. Lin, S. I. Marcus, and W. M. Wonham. Formulas for calculating supremal and normal sublanguages. *Systems and Control Letters*, 15(8):111–117, 1990.

[2] Y. Brave and M. Heymann. On optimal attraction in discrete event processes. Technical Report CIS 9010, Technion - Israel Institute of Technology, Hafia, Israel 32000, 1990.

[3] H. Cho and S. I. Marcus. On supremal languages of class of sublanguages that arise in supervisor synthesis problems with partial observations. *Mathematics of Control Signals and Systems*, 2:47–69, 1989.

[4] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, Reading, MA, 1979.

[5] R. Kumar. *Supervisory Synthesis Techniques for Discrete Event Dynamical Systems.* PhD thesis, The University of Texas at Austin, Austin, Texas 78712, August 1991.

[6] R. Kumar, V. K. Garg, and S. I. Marcus. Supervisory control of discrete event systems: supremal controllable and observable languages. In *Proceedings of 1989 Allerton Conference*, pages 501–510, Allerton, IL, September 1989.

[7] R. Kumar, V. K. Garg, and S. I. Marcus. On controllability and normality of discrete event dynamical systems. *Systems and Control Letters*, 17:157–168, 1991.

[8] R. Kumar, V. K. Garg, and S. I. Marcus. Stability and stabilizability of behavior of discrete event systems. *SIAM Journal of Control and Optimization*, 1991. Submitted.

[9] R. Kumar, V. K. Garg, and S. I. Marcus. Using predicate transformers for supervisory control. In *1991 Conference on Decision and Control*, Brighton, UK, December 1991. To appear.

[10] S. Lafortune and E. Chen. On the infimal closed and controllable superlanguage of a given language. *IEEE Transactions on Automatic Control*, 35(4):398–404, 1990.

[11] E. Lawler. *Combinatorial Optimization - Networks and Matroids.* Holt Rinehart and Winston, 1976.

[12] K. M. Passino and P. J. Antsaklis. Near-optimal control of discrete event systems. In *Proceedings of 1989 Allerton Conference*, pages 915–924, Allerton, IL, September 1989.

[13] K. M. Passino and P. J. Antsaklis. On the optimal control of discrete event systems. In *Proceedings of 1989 IEEE Conference on Decision and Control*, pages 2713–2718, Tampa, FL, December 1989.

[14] P. J. Ramadge and W. M. Wonham. Modular feedback logic for discrete event systems. *SIAM Journal of Control and Optimization*, 25(5):1202–1218, 1987.

[15] P. J. Ramadge and W. M. Wonham. On the supremal controllable sublanguage of a given language. *SIAM Journal of Control and Optimization*, 25(3):637–659, 1987.

[16] P. J. Ramadge and W. M. Wonham. Supervisory control of a class of discrete event processes. *SIAM Journal of Control and Optimization*, 25(1):206–230, 1987.

[17] P. J. Ramadge and W. M. Wonham. The control of discrete event systems. In Y. C. Ho, editor, *Proceedings of IEEE: Special Issue on Discrete Event Systems*, volume 77, pages 81–98. IEEE Press, New York, NY, 1989.

[18] R. Sengupta and S. Lafortune. Optimal control of a class of discrete event systems. In *Proceedings of 1991 IFAC symposium on distributed intelligence systems*, pages 25–30, Arlington, VA, August 1991.

A ROBUST SMITH PREDICTOR WITH ADAPTIVE CONTROL[1]

John T. MacDonald and Geoffrey A. Williamson
Department of Electrical and Computer Engineering
Illinois Institute of Technology
Chicago, Illinois 60616

Summary: This paper addresses the problem of controlling a system with uncertain dynamic characteristics whose output can be measured only after some significant known delay time. Specifically, we would like the actual system output (before the delay) to respond as quickly as possible to command signals. An open loop input schedule is inadequate due to the possible variation in the system dynamics. If simple feedback control of the delayed output is used, loop gains must be reduced to avoid instabilities due to the delay in the feedback. This in turn will limit response time. Another approach is the Smith predictor control in which a model of the plant is run in parallel and the series controller is designed simply to give a desired response in closed loop control of the model [1]. The output computed from the model is used as a nominal feedback signal, and the error between the true plant and the model output (once available) are incorporated into the feedback. This approach works well when the plant and its model have identical dynamic characteristics, but this control configuration is very sensitive to mismodelling of the plant and may become unstable [2].

To resolve the problems associated with mismodelling in the Smith predictor, we use recursive identification from the delayed outputs and equally delayed inputs to form an improved estimate of the plant from the data as it becomes available. This new plant estimate becomes the new parallel model to improve the output prediction. The control law can also be adjusted to give the desired response for the new plant estimate. (See figure 1.) By adjusting the Smith predictor one can improve the system response by attenuating errors in output prediction. If this error can be eliminated, then the speed of response is determined by control of the plant without delay.

The rationale for this approach is predicated on these facts: 1.) If the inputs have a rich information content, then the identification process will reduce the estimate error [3]. 2.) When the plant is stable and the controller is chosen so that the feedback loop with the model is stable, and when the plant and its model are sufficiently close (in the sense of a parameter space metric), the Smith predictor will be stable in the time invariant case [4]. 3.) The identification and adaptation tend to be asymptotically time invariant (if need be, the identification process can be halted).

Simulation studies have demonstrated the potential for improved transient performance in the presence of plant uncertainty using the adapted Smith predictor. We include here a simple example. Figure 2. shows the response of a first order system using a deadbeat control strategy. We compare the response with the model, $P_m(z)$, fixed to the response with P_m adapted using recursive least squares. In this case the initial model deviates only modestly from the plant. It is also possible for the unadapted control to exhibit unstable behavior, while the adapted scheme remains stable. Improved robustness of transient response is shown in figure 3, which displays a region of plant model values satisfying a settling time constraint.

Some questions which remain are: What is the best control design strategy under adaptation given the improving knowledge of the plant dynamics? Under what

---

[1] This work is supported in part by the Whitaker Foundation.

conditions is the identification process stable in this closed loop configuration? How does the transient performance compare with other control schemes such as robust control or other adaptive schemes?

REFERENCES:

[1] O.J.M.Smith, "Closer control of loops with dead time,"Chemical Engineering Progress, Vol.53, pp.217-219, 1957.

[2] I.Horowitz, "Some properties of delayed controls (Smith Regulator)," Int. J. Control, Vol.38, pp.977-990, 1983.

[3] G.C.Goodwin and K.S.Sin, Adaptive Filtering, Prediction, and Control. Englewood Cliffs, NJ: Prentice-Hall Inc., 1984.

[4] Z.J.Palmor and Y.Halevi, "Robustness properties of sampled-data systems with dead time compensators," Automatica, Vol.26, pp.637-640, 1990.

Figure 1. Adapted Smith predictor

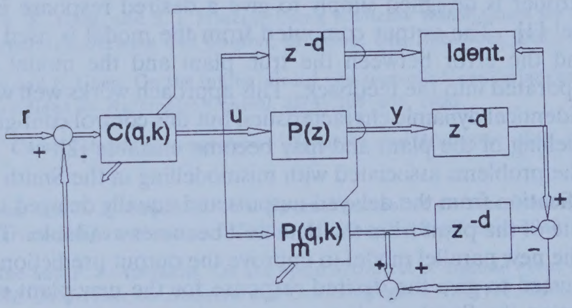


Figure 2. Smith predictor step responses

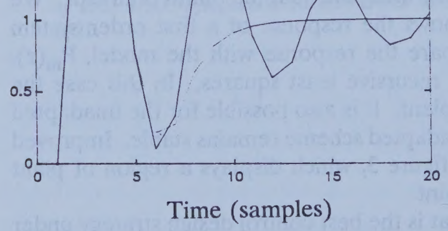
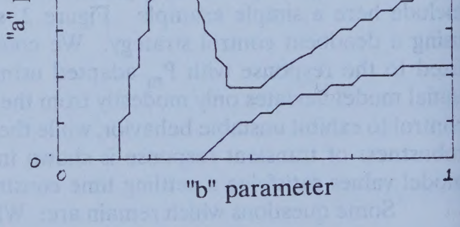


Figure 3. Regions of initial model parameters satisfying a settling time specification



Initial Model

$$P_m(z) = \frac{0.3}{z+0.5}$$

Actual Plant

$$P_m(z) = \frac{0.5}{z+0.5}$$

Initial Plants are varied

$$P_m(z) = \frac{b}{z+a}$$

ON COVARIANCE ASSIGNMENT FOR LINEAR DISCRETE-TIME SYSTEMS

Engin Yaz
Elect. Engr. Dept., University of Arkansas
Fayetteville, AR 72701

Robert E. Skelton
School of Aeronautics & Astronautics
Purdue University, W. Lafayette, IN 47907

Abstract. This paper presents a method of covariance assignment by full-order dynamic compensators for linear discrete-time systems based on noisy output measurements. The closed form solution of the assignment problem is given together with the necessary and sufficient assignability conditions. These conditions are interpreted in light of system theoretical concepts and it is shown that conditions for covariance assignment are not harder to satisfy than those of l_2 designs. Stability robustness of these dynamic feedback schemes is also pointed out for bounded time-varying parameter perturbations and l_2 and l_∞ type input disturbances.

INTRODUCTION

Covariance assignment can be viewed as the stochastic counterpart of the pole assignment control and is presented by Skelton and his co-workers as an alternative to existing scalar-performance-based techniques of control design [1], [2]. This technique is demonstrated to work well for various classes of continuous [1], [3], [4], [8], and discrete systems [2], [5] - [8]. This control scheme has several important advantages like control over transient response through Markov parameters [9], control of the effects of roundoff errors in digital control [10], and adjustability of residence time through the maximum eigenvalue of the state covariance in aiming control problems [11]. The biggest advantage is the fact that this controller does not require the solution of any Riccati or Lyapunov type equations and is given in closed form.

The present work will concentrate on the dynamic covariance control of linear discrete time systems using full order controllers. The basic model used in [5] ignored the effects of measurement noises which will be included here. In the next section, both the assignable covariances and the dynamic compensators that achieve covariance assignment are characterized. We employ a different approach from the singular value decomposition used in [5], which allows us to interpret existence conditions using system theoretic concepts. By these means, we show that the existence conditions for covariance assignment are about as easy to satisfy as those of LQG regulators. Then, we present a robustness property of these controllers which is the maintenance of state boundedness in the presence of time-varying parameter variations, l_2, and l_∞ type disturbance inputs. We also indicate some possible generalizations of this work in the discussion section.

We fix the notation at this point for the following development. $\mathbb{R}^n$ is the n-dimensional real Euclidean vector space with norm $\|x\| = (x^T x)^{1/2}$, x^T denoting the transpose of x. $\mathbb{R}^{nxm}$ is the set of all (nxm)-dimensional matrices with real elements. $\|A\|_s$ is the spectral norm (the one induced by the Euclidean vector norm) of matrix A. For symmetric A, $\bar{\lambda}(A)$ ($\underline{\lambda}(A)$) denotes the maximum (minimum) eigenvalue of A. For symmetric matrices A, B, A > B (A ≥ B) means A -B is a positive (non-negative) definite matrix. A^+ will denote the unique Moore-Penrose pseudo-inverse [12] and $A^{1/2}$ is a square root (non-unique, in general) of A. l_2 denotes the space of all infinite vector sequences $\{x_0, x_1, \dots\}$ for which the norm $(\sum_{i=o}^{\infty} \|x_i\|^2)^{1/2}$ is finite. l_∞ is the space of all bounded vector sequences $\{x_i, i \geq 0\}$ for which $\sup_{i \geq 0} \|x_i\| < \infty$. $E\{.\}$ denotes the expectation operator.

COVARIANCE ASSIGNMENT

Consider the linear time-invariant discrete time system

$$x^s_{k+1} = A^s x^s_k + B^s u_k + v_k \tag{1}$$

and the measurement equation

$$y_k = C^s x^s_k + w_k \tag{2}$$

where $x^s_k \in \mathbb{R}^n$ is the state, $u_k \in \mathbb{R}_m$ the control input, and $y_k \in \mathbb{R}^p$ is the measured output. v_k and w_k are zero mean white noise sequences with respective covariances Σ_v and Σ_w which are mutually uncorrelated and both independent of the initial state x^s_o that has mean $\bar{x}^s_o$ and covariance P_0. Coefficient matrices A^s, B^s, and C^s are all constant and of appropriate dimensions. To produce the control action u_k for system (1) based on the measurements y_k, we use the time-invariant linear dynamic compensator

$$x^c_{k+1} = A^c x^c_k + B^c y_k \tag{3a}$$

$$u_k = C^c x^c_k + D^c y_k \tag{3b}$$

with $x^c_k \in \mathbb{R}^n$ that results in the closed loop system

$$x_{k+1} = A_{CL} x_k + B_{CL} \zeta_k \tag{4}$$

where

$$x_k = \begin{bmatrix} x^s_k \\ x^c_k \end{bmatrix}, A_{CL} = \begin{bmatrix} A^s + B^s D^c C^s & B^s C^c \\ B^c C^s & A^c \end{bmatrix}, B_{CL} = \begin{bmatrix} I & B^s D^c \\ 0 & B^c \end{bmatrix}, \zeta_k = \begin{bmatrix} v_k \\ w_k \end{bmatrix} \tag{5}$$

We define the steady state value of the second moment of closed loop system state as follows

$$X = \lim_{k\to\infty} E\{x_k x_k^T\} = \begin{bmatrix} X^1 & X^{12} \\ (X^{12})^T & X^2 \end{bmatrix} \tag{6}$$

which satisfies

$$X = A_{CL} X A_{CL}^T + B_{CL} V B_{CL}^T \tag{7}$$

where V = block diag [Σ_v, Σ_w] if the steady state exists.

Our aim in this paper is twofold: First, to determine if a given $X = X^T > 0$ is assignable by the compensator (3a) - (3b), and, second, if X is assignable, to compute the coefficient matrices A^c, B^c, C^c, and D^c of the compensator to assign this X. To achieve these aims, we rewrite (7) as

$$X = (A + BKC)X(A + BKC)^T + (I_{11} + BKI_{12})\, V(I_{11} + BKI_{12})^T \tag{8}$$

where

$$A = \begin{bmatrix} A^s & 0 \\ 0 & 0 \end{bmatrix},\ B = \begin{bmatrix} B^s & 0 \\ 0 & I \end{bmatrix},\ K = \begin{bmatrix} D^c & C^c \\ B^c & A^c \end{bmatrix},\ C = \begin{bmatrix} C^s & 0 \\ 0 & I \end{bmatrix}$$
$$I_{11} = \begin{bmatrix} I_n & 0 \\ 0 & 0 \end{bmatrix},\ I_{12} = \begin{bmatrix} 0 & I_p \\ 0 & 0 \end{bmatrix} \tag{9}$$

Rearranging (8), we obtain

$$\begin{aligned} X - AXA^T - I_{11}VI_{11} &= BK(CXA^T + I_{12}VI_{11}) + (AXC^T + I_{11}\, VI_{21})K^TB^T \\ &\quad + BK(CXC^T + I_{12}\, VI_{21})K^TB^T \end{aligned} \tag{10}$$

where $I_{21} = I_{12}^T$ and $I_{11}^T = I_{11}$.
Completing the square on the right side, we have

$$\begin{aligned} X - AXA^T - I_{11}VI_{11} + Y(CXC^T + I_{12}VI_{21})Y^T \\ = (BK + Y)(CXC^T + I_{12}VI_{21})\,(BK + Y)^T \end{aligned} \tag{11}$$

with

$$\begin{aligned} Y &= (AXC^T + I_{11}VI_{21})\,(CXC^T + I_{12}VI_{21})^{-1} \\ &= AXC^T(CXC^T + I_{12}VI_{21})^{-1} \end{aligned} \tag{12}$$

assuming that the inverse exists. The simplification in (12) is possible due to the fact that v_k and w_k are uncorrelated. Let us now see under what conditions this inverse exists. Using the definitions of C, X, I_{12}, and V, we get

$$CXC^T + I_{12}\, VI_{21} = \begin{bmatrix} C^s X^1 (C^s)^T + \Sigma_w & C^s X^{12} \\ (X^{12})^T C^{sT} & X^2 \end{bmatrix} \tag{13}$$

To guarantee the positive definiteness of this partitioned matrix, we need to have [13], $X^2 > 0$ and $C^s[X^1 - X^{12}(X^2)^{-1}(X^{12})^T](C^s)^T + \Sigma_w > 0$.. Since, the positive definiteness of X or

$$X = \begin{bmatrix} X^1 & X^{12} \\ (X^{12})^T & X^2 \end{bmatrix} > 0 \tag{14}$$

guarantees that $X^2 > 0$ and $X^1 - X^{12}(X^2)^{-1}\,(X^{12})^T > 0$, $\Sigma_w > 0$ is sufficient to guarantee the existence of the inverse in (12).

Let us now assume that the left side of equation (11) is nonnegative definite and is of maximum rank (p+n), and rewrite (11) as

$$LL^T = (BK+Y)TT^T\,(BK+Y)^T \tag{15}$$

where $L \epsilon \mathbb{R}^{2n x (p+n)}$ and $T \epsilon \mathbb{R}^{(p+n) x (p+n)}$ are respectively the square root matrices for the left side of (11) and $CXC^T + I_{12} VI_{21}$. It follows from (15) that for an orthogonal matrix U, it is true that

$$LU = (BK + Y)T \tag{16}$$

Since T is nonsingular which follows from the same property of $(CXC^T + I_{12}\ VI_{21})$, we can write

$$BK = LUT^{-1} - Y \tag{17}$$

with Y given by (12). Our purpose is to find K, and K can be found [12] as

$$K = B^+\,(LUT^{-1} - Y) + (I - B^+B)Z \tag{18}$$

where Z is arbitrary, if and only if

$$BB^+\,(LUT^{-1} - Y) = (LUT^{-1} - Y) \tag{19}$$

which is called the "consistency" condition. We can now sum up our findings as follows:

Theorem 1. A given $X = X^T > 0$ is assignable by the compensator (3a) - (3b) for the system (1), if $\Sigma_w > 0$, the left side of equation (11) is nonnegative definite, is of maximum rank (p + n), and (19) holds. If X is assignable, then the compensator that accomplishes this is given by (3a)-(3b) with its parameters determined by (18).

ON EXISTENCE CONDITIONS

Although the two questions we posed earlier on the assignability and how to assign are answered by theorem 1, the assignability conditions require further clarification. Especially, it is important to relate these conditions to commonly used system theoretical concepts like stabilizability, detectability, etc. to see if dynamic output covariance assignment requires more stringent conditions than for example, LQG regulators. For this purpose, we consider the condition of nonnegativity of the left side of equation (11). Substituting from (6), (9), and (12) into (11) and after a few manipulations involving the inverse of partitioned matrices, we find that the nonnegativity condition boils down to

$$\begin{bmatrix} \tilde{X} & X^{12} \\ (X^{12})^T & X^2 \end{bmatrix} \geq 0 \tag{20}$$

with

$$\tilde{X} = X^1 - A^s\bar{X}(A^s)^T + A^s\bar{X}(C^s)^T\,[C^s\bar{X}(C^s)^T + \Sigma_w]^{-1}\,C^s\bar{X}(A^s)^T - \Sigma_v \tag{21}$$

where $\bar{X} = X^1 - X^{12}(X^2)^{-1}(X^{12})^T$. Since X > 0, both X^2 and $\bar{X}$ are positive definite. So, to satisfy (20), we only need to have [13]:

$$\tilde{X} - X^{12}(X^2)^{-1}\,(X^{12})^T \geq 0 \tag{22}$$

Using (21), this simply means that the Riccati matrix inequality

$$\bar{X} \geq A^s\,\bar{X}(A^s)^T - A^s\bar{X}(C^s)^T\,[C^s\bar{X}(C^s)^T + \Sigma_w]^{-1}\,C^s\bar{X}(A^s)^T + \Sigma_v \tag{23}$$

has to have a solution $\bar{X} > 0$. The conditions for the existence of more general inequalities than the present one are found in [13] and [14]. If $(A^s, \Sigma_v^{1/2})$ is controllable, then the detectability of the pair (A^s, C^s) is necessary and sufficient for the existence of a solution $\bar{X} = \overline{X^T} > 0$ to (23). So, one can see that nonnegativity condition for (11) can be expressed in terms of some well-known system theoretical concepts.

Let us now look at condition (19). Rewriting this, we manipulate it as follows:

$$(I - BB^+)\,LU = (I - BB^+)YT \tag{24}$$

and upon squaring

$$(I - BB^+)\,[LL^T - Y(CXC^T + I_{12}VI_{21})Y^T]\,(I - BB^+) = 0 \tag{25}$$

Using (11) and (15),

$$(I - BB^+)\,[X - AXA^T - I_{11}VI_{11}]\,(I - BB^+) = 0 \tag{26}$$

and from the definitions of the variables involved, [26] is equivalent to

$$\begin{bmatrix} [I - B^s(B^s)^+]\ [X^1 - A^s X^1 (A^s)^T - \Sigma_v]\ [I - B^s(B^s)^+] & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{27}$$

By the properties [12]:

$$(I - BB^+)\, B = 0\ ,\ B^T(I - BB^+) = 0 \tag{28}$$

of the pseudo-inverse, (27) can be written for an arbitrary $G \epsilon \mathbf{R}^{mxn}$ as

$$[I - B^s(B^s)^+][X^1 - (A^s + B^s G)X^1(A^s + B^s G)^T - (B^s V_1 B^{sT} + V_2)][I - B^s(B^s)^+] = 0 \tag{29}$$

where Σ_v can always be decomposed like this with symmetric, nonnegative definite $V_1 \epsilon \mathbf{R}^{mxm}$, and $V_2 \epsilon \mathbf{R}^{nxn}$, so that $\Sigma_v = \Sigma_v^{1/2}\ \Sigma_v^{T/2} = (B^s V_1^{1/2},\ V_2^{1/2})\ (B^s V_1^{1/2},\ V_2^{1/2})^T$, and $V_1 > 0$. Now, one can show that $(A^s + B^s G,\ [B^s V_1^{1/2},\ V_2^{1/2}])$ is controllable if $(A^s,\ V_2^{1/2})$ is controllable by using Popov-Belevitch-Hautus test [15]: for all z, we have

$$n = \text{rank}\ [zI - A^s,\ V_2^{1/2}] \tag{30}$$

So, considering the Lyapunov difference equation

$$X_{k+1}^1 = (A^s + B^s G)X_k^1\ (A^s + B^s G)^T + \Sigma_v \tag{31}$$

we know that if (A^S, B^S) is stabilizable, and G is chosen such that $(A^S + B^S G)$ has its eigenvalues inside the unit circle and $(A^s + B^s G,\ \Sigma_v^{1/2})$ is controllable or equivalently $(A^s,\ V_2^{1/2})$ is controllable as is shown, then a unique positive definite solution X^1 which satisfies the algebraic version of (31) will exist. In this case, condition (29) will be satisfied because the expression in the middle square brackets is zero.

Note that controllability of $(A^s,\ V_2^{1/2})$ will also imply the controllability of $(A^s,\ \Sigma_v^{1/2})$ which was one of the conditions obtained to guarantee the satisfaction of the nonnegativity condition in (11).

We again summarize our basic results below:

Theorem 2. Let $\Sigma_w > 0$, and $(A^s,\ V_2^{1/2})$ be controllable with $V_2 = \Sigma_v - B^s V_1 B^{s^T}$ for a nonnegative definite V_1. If (A^S, B^S) is stabilizable and (A^S, C^S) is detectable, then there exist covariances (second moments) $X = X^T > 0$ which satisfy the nonnegativity condition of (11) and consistency condition (19), which are assignable by the dynamic compensator (3a) - (3b) to system (1).

One can immediately recognize that the existence conditions are no more stringent than the ones used in LQG regulator design.

ROBUSTNESS

In this section, we are going to investigate the stability robustness of dynamic covariance controllers in the presence of possibly time-varying but bounded parameter perturbations and unmodelled l_2 and l_∞ type external disturbances. So, suppose that we are dealing with a perturbed system

$$z_{k+1} = (A^s + \Delta A_k^s)z_k + (B^s + \Delta B_k^s)u_k + \zeta_k^1 \tag{32a}$$

$$y_k = (C^s + \Delta C_k^s)z_k + \zeta_k^2 \tag{32b}$$

where we have both time-varying (bounded) parameter variations and also l_2 or l_∞ type disturbances ζ_k^1, ζ_k^2 which are not taken into consideration in design which is based on the nominal system and measurement descriptions (1) and (2). We have shown that under some mild conditions (on the nominal model), a chosen covariance $X = X^T > 0$ may be assigned that satisfies (7):

$$X = A_{CL} X A_{CL}^T + B_{CL} V B_{CL}^T \tag{7}$$

Obviously, this means that the dynamic output feedback covariance controller is stabilizing for the system or A_{CL} has all of its poles strictly inside the unit circle. Therefore one can find a sufficiently small $\alpha > 0$ such that

$$\|A_{CL}^k\|_s < \beta \ (1 + \alpha)^{-k/2} \tag{33}$$

for some $\beta \geq 1$. Since a matrix and its transpose have the same set of eigenvalues, A_{CL}^T will also satisfy (33). This implies from the theory of Lyapunov equations that there exists a solution
$\chi = \chi^T \geq 0$ to

$$\chi = (1 + \alpha) A_{CL}^T \chi A_{CL} + B_{CL} V B_{CL}^T \tag{34}$$

Moreover, for the purpose of robustness analysis, we will assume that $B_{CL} V B_{CL}^T > 0$, which results in the uniqueness of the positive definite χ. At this point, we introduce an auxilary function $W_k = \bar{z}_k^T \chi \bar{z}_k > 0$, $\bar{z}_k = [z_k^T, (x_k^c)^T]^T$,, which will help us investigate the stability robustness properties like a Lyapunov function.
By defining

$$\Delta A_{CL} = \begin{bmatrix} \Delta A_k^s + \Delta B_k^s D^c (C^s + \Delta C_k^s) + B^s D^c \Delta C_k^s & \Delta B_k^s C^c \\ B^c \Delta C_k^s & 0 \end{bmatrix}, \quad \Delta B_{CL} = \begin{bmatrix} 0 & \Delta B_k^s D^c \\ 0 & 0 \end{bmatrix} \tag{35}$$

$$\Xi_k = \begin{bmatrix} \zeta_k^1 \\ \zeta_k^2 \end{bmatrix}$$

letting $\|\Delta A_{CL}\|_s \leq b_A$ and $\|\Delta B_{CL}\|_s \leq b_B$, assuming that b_A is small enough so that

$$\underline{\lambda}(B_{CL} V B_{CL}^T) - (1 + \alpha)\bar{\lambda}(\chi)\, b_A(2\|A_{CL}\|_s + b_A) \triangleq \epsilon > 0 \tag{36}$$

and using $\underline{\lambda}(\chi)\,\|\bar{z}_k\|^2 \leq W_k \leq \bar{\lambda}(\chi)\,\|\bar{z}_k\|^2$, we get

$$\|\bar{z}_k\|^2 \leq \left(1 - \frac{\epsilon}{\bar{\lambda}(\chi)}\right)^k \frac{\bar{\lambda}(\chi)}{\underline{\lambda}(\chi)} \|\bar{z}_o\|^2 + (1 + \alpha^{-1}) \frac{\bar{\lambda}(\chi)}{\underline{\lambda}(\chi)} (\|B_{CL}\|_s + b_B)^2$$
$$\sum_{i=o}^{k-1} \left(1 - \frac{\epsilon}{\bar{\lambda}(\chi)}\right)^{k-i-1} \|\Xi_i\|^2 \tag{37}$$

It is easy to show that $0 < \left(1 - \dfrac{\epsilon}{\bar{\lambda}(\chi)}\right) < 1$. If Ξ_i is an l_2 type disturbance such that $\sum_{i=0}^{\infty} \|\Xi_i\|^2 \leq b_\Xi^2 < \infty$, then

$$\lim_{k\to\infty} sup\; \|\bar{z}_k\|^2 \leq \frac{(1 + \alpha^{-1}).\bar{\lambda}^2(\chi)}{\epsilon\; \underline{\lambda}(\chi)} (\|B_{CL}\|_s + b_B)^2 \cdot b_\Xi^2 \tag{38}$$

If Ξ_i is an l_∞ type disturbance so that $\sup_{k \geq o} \|\Xi_i\| \leq b_\Xi^\infty < \infty$,

$$\lim_{k\to\infty} sup\, \|\bar{z}_k\|^2 \leq (1 + \alpha^{-1}) \frac{\bar{\lambda}^2(\chi)}{\underline{\lambda}(\chi)} (\|B_{CL}\|_s + b_B)^2 \cdot (b_\Xi^\infty)^2 \tag{39}$$

We sum these results as follows:

Theorem 3. The compensator (3a) - (3b) which is designed based on the nominal model (1) and (2), maintains the boundedness of the perturbed state (32a) - (32b) if $B_{CL} V B_{CL}^T > 0$ and the parameter perturbations do not exceed the magnitudes specified by (36). The magnitude bounds on the perturbed states are given by (38) and (39) respectively for l_2 and l_∞ type disturbances.

The above theorem states that the covariance assignment controller has the desired properties of stability robustness for parameter perturbations and disturbance reduction for l_2 and l_∞ type process and measurement disturbances. The above results should be interpreted as qualitative rather than quantitative results because - although explicit expressions are given - it may be tedious to calculate these bounds in a particular design situation.

DISCUSSION AND FURTHER EXTENTIONS

The main contribution of this work is to extend the use of dynamic covariance assignment controllers for discrete systems to the case where there is measurement noise. In our derivation, we assumed that process and measurement noises are uncorrelated for the sake of presentation simplicity. However, this needs not be the case. For example, if $E\{v_k w_k^T\} = \Sigma_{vw}$, then there are only a few minor changes in the design equations such as in (12), $I_{11}\ VI_{21}$ term will be retained and its value will be

$$I_{11}\ VI_{21} = \text{block } diag\ [\Sigma_{vw}, 0]$$

The inequality (23) will become

$$\overline{X} \geq A^s \overline{X} (A^s)^T - [A^s \overline{X} (C^s)^T + \Sigma_{vw}]\ [C^s \overline{X} (C^s)^T + \Sigma_w]^{-1} (C^s \overline{X} (A^s)^T + \Sigma_{vw}^T) + \Sigma_v$$

and the rest of the development will be correct. In addition to mutually correlated noises, one can also have autocorrelated noises, in which case dynamic models of these noises should be incorporated into the state model [16].

Another contribution of this paper is in its slightly different approach to covariance control design which allows us to illustrate that the existence conditions are no more restrictive than those used in LQG regulators. This coupled with the fact that one does not need to solve any Riccati or Lyapunov equations are what make the covariance control advantageous over the existing design schemes like LQG (l_2) and H^∞, etc.

Still another point that becomes apparent due to the use of a different approach in covariance assignment resulting in inequality (23) is that if $(A^s, \Sigma_v^{1/2})$ is controllable, and (A^S, C^S) detectable, then one can find an $\overline{X} = \overline{X}^T > 0$ (not necessarily the one we want) that can be assigned. Denoting the right side of (23) by $\Phi(\overline{X})$, we rewrite (23) as

$$\overline{X} \geq \Phi(\overline{X}) \tag{23}$$

Now, one can use the results on generalized Riccati equations in [14], to conclude that the matrix sequence defined by

$$\overline{X}_k = \Phi^k(\overline{X})$$

has a monotonically decreasing behavior, that is

$$\overline{X}_k \geq \overline{X}_{k+1}$$

Since this sequence is lower bounded by $\Sigma_v \geq 0$ and all elements of the sequence are positive definite, it converges to $\overline{X}_{ss} > 0$ that satisfies

$$\overline{X}_{ss} = \Phi(\overline{X}_{ss})$$

that is, the unique positive definite solution of the algebraic Riccati equation. This implies that

$$\overline{X} \geq \overline{X}_1 \geq \overline{X}_2 \geq \dots \geq \overline{X}_{ss}$$

or that $\overline{X}$ satisfying (23) has to be larger (in the sense of this particular partial ordering) than the corresponding unique positive definite solution of the Riccati equation. This establishes a natural lower bound on the assignable covariances $\overline{X} = X^1 - X^{12}(X^2)^{-1}(X^{12})^T$.

REFERENCES

[1] A. Hotz and R. E. Skelton, "Covariance control theory," Int. J. Contr., vol. 46, pp. 13-32, 1987.

[2] E. G. Collins and R. E. Skelton, "A theory of state covariance assignment for discrete systems," IEEE TAC, vol. 32, pp. 35-41, 1987.

[3] R. E. Skelton and M. Ikeda, "Covariance controllers for linear continuous time systems," Int. J. Contr., vol. 49, pp. 1773-1785, 1989.

[4] Y. Yasuda, S. M. Kherat, R. E. Skelton, and E. Yaz, "Covariance control and robustness of bilinear systems," Proc. of CDC, Honolulu, Hawaii, pp. 1421-1425, 1990.

[5] C. Hsieh and R. E. Skelton, "All covariance controllers for linear discrete-time systems," IEEE TAC, vol. 35, pp. 908-915, 1990.

[6] R. E. Skelton, S. M. Kherat, and E. Yaz, "Covariance control of discrete stochastic bilinear systems," Proc. of ACC, Boston, MA, pp. 2660-2664, 1991.

[7] E. Yaz, "State covariance assignment for discrete time systems with Jump Markov parameters," Proc. of European Contr. Conf., Grenoble, France, July 1991. (to appear).

[8] E. Yaz and R. E. Skelton, "Continuous and discrete state estimation with error covariance assignment," Proc. of 1991 CDC, Brighton, England (to appear).

[9] R. E. Skelton, Dynamic Systems Control, New York: Wiley, 1988.

[10] D. Williamson, "Roundoff noise minimization and pole-zero sensitivity in fixed point digital filters using residue feedback," IEEE Trans. ASSP, vol. 34, pp. 1013-1016, 1986.

[11] S. Meerkov and T. Runalfson, "Residence time control," IEEE TAC, vol. 33, pp. 323-332, 1988.

[12] A. Ben-Israel and T.N.E. Greville, Generalized Inverse Theory and Applications, New York: Wiley, 1974.

[13] E. Yaz, "Equivalence of two stochastic stabilizability conditions and its implications," Int. J. Systems Sci., vol. 20, pp. 1745-1751, 1989.

[14] --, "On the stabilizing control of a class of nonlinear stochastic systems," IEEE TAC, vol. 36, pp. 121-123, 1991.

[15] T. Kailath, Linear Systems, Englewood Cliffs, N.J.: Prentice-Hall, 1980.

[16] B. D. O. Anderson and J. B. Moore, Optimal Filtering, Englewood Cliffs, N.J.: Prentice-Hall, 1979.

NONPARAMETRIC DETECTION AND MEASUREMENT OF DISCONTINUITIES

Miroslaw Pawlak

Department of Electrical and Computer Engineering
The University of Manitoba
Winnipeg, Manitoba, Canada R3T 2N2

ABSTRACT

We study the detection and size measurement of discontinuities from sampled noisy data of a function. Estimates of a discontinuity size stemming from the theory of Fourier series are proposed.

The convergence properties (consistency and rate of convergence) of the proposed estimates are established. Our asymptotic results model the performance of the estimates on grids which become increasingly fine, i.e., as a pixel size tends to zero.

I. PRELIMINARIES

Consider the regression model

$$g(x_i) = f(x_i) + \varepsilon(x_i), \text{ i. } = 1, 2, \ldots, n , \qquad (1)$$

where $x_1, x_2, \ldots, x_n$ are known design points in [a, b], $\varepsilon(x_1), \ldots, \varepsilon(x_n)$ are zero mean, uncorrelated random errors having common variance σ^2, and f is an unknown function that we wish to estimate. There are many effective nonparametric methods of estimating f including smoothing splines, kernel, and orthogonal series estimators [1]. None of these techniques is bearing the information that the function might contain discontinuity points. The pretence of change points in an otherwise smooth function is very common in problems of signal processing and computer vision [2], [3]. In fact, discontinuities (edges) play a crucial role in early vision where changepoints represent a major component of image reconstruction, segmentation, and interpretation [3]. A number of techniques have been proposed for localization of discontinuity points in the underlying function. Rarely, however they have been based on the theoretical justification, with the exception of [4], [5], [6].

In [5], [6] a simple parametric model of a discontinuous univariate function has been assumed. That is, if f has a changepoint at $\theta \in (a,b)$, i.e., $f(\theta+) \neq f(\theta-)$, then f(x) is identical to different horizontal line segments in $[a, \theta]$ and $[b, \theta]$, respectively. Furthermore, it has been claimed [5], [6] that "almost no effort has been made for the measurement of the size of discontinuities", i.e., to estimate $f(\theta+) - f(\theta-) = \Delta(\theta)$, say. We argue in this paper that such a problem was addressed in the early 20th century in the relation to the so-called Gibbs phenomenon [7], [9], [10]. There, it was elaborated in the context of convergence of the partial sums of Fourier series in the neighborhood of a discontinuity of the function being expanded. Here we utilize this approach for estimating the discontinuity size $\Delta(\theta)$ and location θ of f, when the model in (1) is assumed.

No specific parametric form of f is postulated and in a consequence a nonparametric regression estimation technique based on Fourier expressions is proposed.

II. ESTIMATES AND CONVERGENCE RESULTS

The following two estimates of $\Delta(x)$ - the discontinuity size at $x \in (-\pi, \pi)$ are examined.

$$\widehat{\Delta}_F(x) = \frac{\pi}{N} \sum_{k=1}^{N} k\,(\widehat{b}_k \cos kx - \hat{a}_k \sin kx) ,$$

$$\widehat{\Delta}_L(x) = \frac{\pi}{\ln N} \sum_{k=1}^{N} (\widehat{b}_k \cos kx - \hat{a}_k \sin kx) ,$$

where $\hat{a}_k$, $\widehat{b}_k$ are estimates of the Fourier coefficients of f(x) defined as

$$\hat{a}_k = \frac{2 \sin\left(k\,\delta_n/2\right)}{\pi\,k} \sum_{i=1}^{n} g(x_i) \cos(k\,x_i) ,$$

$$\widehat{b}_k = \frac{2 \sin\left(k\,\delta_n/2\right)}{\pi\,k} \sum_{i=1}^{n} g(x_i) \sin(k\,x_i) .$$

Here $\delta_n = x_i - x_{i-1}$ is the pixel size, N is the truncation parameter. The estimate $\widehat{\Delta}_F(x)$ stems from the Fejer technique [10], while $\widehat{\Delta}_L(x)$ is due to Lukacs [9].

It is proved that both $\widehat{\Delta}_F(x)$ and $\widehat{\Delta}_L(x)$ tend to $\Delta(x)$ (in the mean square sense) as $N\,\delta_n \to 0$ and $N \to \infty$ for $n \to \infty$, see [8] for further details.

The rate of convergence for $\widehat{\Delta}_F(x)$ is of order $n^{-2/3}$, while $\widehat{\Delta}_L(x)$ has an extremely slow rate of order 1/ln n. Nevertheless, we argue that the $\widehat{\Delta}_L(x)$ detector can have a better localization power than $\widehat{\Delta}_F(x)$.

Thus, $\widehat{\Delta}_F(x)$, $\widehat{\Delta}_L(x)$ converge to $\Delta(\theta)$ at the discontinuity point $x = \theta$, and they tend to zero at any point of continuity. That is, one can use extrema in $\widehat{\Delta}_F(x)$ and $\widehat{\Delta}_L(x)$ as a position of a discontinuity. The obtained convergence results assure that for a sufficiently small pixel size and properly selected truncation parameter N we can get the successful detection of discontinuities.

REFERENCES

[1] R.L. Eubank (1988), Spline Smoothing and Nonparametric Regression, New York: Marcel-Dekker.

[2] M. Basseville (1988), "Detecting changes in signals and systems - a survey", Automatica, vol. 24, pp. 309-326.

[3] B.K.P. Horn (1986), Robot Vision, The MIT Press, Massachusetts.

[4] J. Canny (1987), "Computational approach to edge detection", IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 8, pp. 679-698.

[5] D. Lee (1990), "Coping with discontinuities in computer vision: their detection, classification and measurement", IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 12, pp. 321-344.

[6] D. Lee (1991), "Detection, classification and measurement of discontinuities", SIAM J. Sci. Stat. Comput., vol. 12, pp. 311-341.

[7] A. Zygmund (1959), Trigonometric Series, vols. 1, 2, Cambridge University Press, Cambridge.

[8] M. Pawlak (1991), "On the measurement and detection of discontinuities", Technical Report, presented also at the IEEE Symp. on Inf. & Theory,1991.

[9] F. Lukacs (1920), "Uber die Bestimmung des Sprunges einer Funktion aus ihrer Fourierreihe", J. Reine Anger. Math., vol. 150, pp. 107-112.

[10] L. Fejer (1913), "Uber die Bestimmung des Sprunges der Funktion aus ihrer Fourierreihe", J. Reine Anger. Math., vol. 142, pp. 165-188.

DYNAMIC DISTURBANCE REJECTION CONTROL FOR DISCRETE NONLINEAR STOCHASTIC SYSTEMS

A. Azemi and E. Yaz
Electrical Engineering Department
University of Arkansas, Fayetteville, AR 72701

ABSTRACT

A dynamic control technique is given which minimizes the effect of process and measurement disturbances for a class of discrete-nonlinear stochastic systems. A Lyapunov based nonlinear filter that was developed previously by the authors is utilized for the estimation part required in this disturbance minimization technique.

MAIN RESULTS

The minimization of the effects of disturbance having known waveform and acting on a linear system has been treated in [1] and [2]. This concept was extended in [3], [4], and [5] to nonlinear systems where the disturbances themselves could possess nonlinear system models. [6] considers a larger class of nonlinearities and generalizes the results of [3] and [4]. This technique can be viewed as a possible alternative to controllers with disturbance robustness properties such as the H^∞ design [7] which minimizes a certain scalar measure of sensitivity over all frequencies for a specified disturbance description. Other well-known design methods are those of [8] and [9] which can accommodate unknown but bounded disturbances and l^1 controllers of [10] which can guarantee a given bound on the response for a specific disturbance. In this work, we will introduce a dynamic controller that is capable of minimizing the effect of known-waveform disturbances in both state and measurement equations, which are allowed to be noise-driven. This technique does not require the knowledge of either the magnitude or the arrival time of the disturbance. The information about the "disturbance state" is indirectly obtained via construction of a nonlinear estimator proposed in [11].

Let us consider the signal model given by the following discrete-time nonlinear stochastic difference equation :

$$x_{k+1} = Ax_k + Bu_k + \phi_s(x_k) + F_1 w_k + g_1(x_k) v_k^1 \tag{1}$$

and the measurement model :

$$y_k = Cx_k + \phi_m(x_k) + F_2 w_k + g_2(x_k) v_k^2 \tag{2}$$

where $x_k \in \mathbf{R}^n$ is the state $u_k \in \mathbf{R}^m$ is the deterministic input, and $y_k \in \mathbf{R}^p$ is the observed output of the system. $v_k^1 \in \mathbf{R}^{q_1}$ and $v_k^2 \in \mathbf{R}^{q_2}$ are uncorrelated standard white noise vectors. $\phi_s(x_k)$ and $\phi_m(x_k)$ are nonlinear functions of x_k, with $\| \phi_m(x_k) \| \leq \beta_\phi$, a positive constant for all $x_k \in \mathbf{R}^n$. $g_1(x_k)$ and $g_2(x_k)$ represent the state dependent noise intensities and satisfy the following bounds :

$$\| g_1(x_k) \|_s \leq \beta_1 \quad , \quad \| g_2(x_k) \|_s \leq \beta_2 \tag{3}$$

where β_1, β_2 are finite positive constants. $w_k \in \mathbf{R}^r$ is a disturbance with a known waveform (step, ramp, sinusoid, etc.) but unknown magnitude and arrival time. We assume that w_k can not be measured and it satisfies the following relation :

$$w_{k+1} = Dw_k + \phi_d(x_k) + \sigma_k \tag{4}$$

where $\phi_d(x_k)$ is the nonlinearity dependent on x_k and σ_k is a "sparsely populated" sequence of unknown impulses [2].

First, we will form the pseudo-output as follows :

$$\bar{y}_k \;=\; Gy_k + Eu_{k-1} \tag{5}$$

by using the previous control action with $u_{-1} = 0$.

To realize disturbance compensation, we will be using part u_k^d of the control $u_k = u_k^c + u_k^d$ whereas the other part, u_k^c can be used for the purpose of accomplishing the main task of the control such as stabilization.

The following disturbance minimization controller is used :

$$u_k^d = -\underline{B}^{\#}\underline{F}\hat{w}_k \tag{6}$$

where

$$\underline{B} = \begin{bmatrix} B \\ GCB + E \end{bmatrix}, \quad \underline{F} = \begin{bmatrix} F_1 \\ G(CF_1 + F_2D) \end{bmatrix} \tag{7}$$

and $\hat{w}_k$ is the estimate of w_k obtained from a convergent estimator.

But, first we will assume that w_k can be estimated perfectly, and present conditions under which the control (6) can completely reject any known-waveform disturbance w_k :

Theorem. Suppose that w_k can perfectly be estimated in an asymptotic sense, and B is of full rank. Then, control (6) can completely eliminate both process and measurement disturbances if and only if

$$(I - BB^{\#})F_1 = 0 \ , \ GF_2D = EB^{\#}F_1 \tag{8}$$

Then, the Lyapunov-based estimator proposed in [11] is used to produce the estimates w_k needed in (6). It can be theoretically shown that under some mild conditions, the estimator is guaranteed to have a mean square exponentially and almost surely ultimately bounded estimation error. The simulation results omitted, due to lack of space here, confirm this finding.

REFERENCES

[1] C. D. Johnson, "Accommodation of external disturbance in linear regular and servomechanism problem," *IEEE Trans. Automatic Contr., vol. Ac-16, pp. 635-644, 1971.*

[2] C. D. Johnson, "Disturbance accommodation control : An overview of the subject," *J. Interdis. Model and Simul., vol. 3, pp. 1-29, 1980.*

[3] E. Yaz, "Accommodation for disturbance having nonlinear models and acting on nonlinear systems," *IEEE Trans. on Circuits and Systems, vol. CAS-33, No. 3, pp. 342-343, 1986.*

[4] E. Yaz and T. W. Martin, "Disturbance minimization for nonlinear systems acted upon by disturbances having nonlinear models," *Proc. 29th Midwest Symposium of Circuits and Systems, Lincoln, Nebraska, pp. 728-731, 1986.*

[5] C. D. Johnson, "Disturbance accommodating control for nonlinear systems," *Proc. 18th Southeastern Symp. Syst. Theory, Washington D.C., pp. 373-375, 1986.*

[6] E. Yaz and T. W. Martin, "Generalization of disturbance minimization scheme," *Proc. 8th ACC, vol. 3, pp. 2580-2585, 1989.*

[7] B. A. Francis, *A Course in H^∞ Control Theory,* Spinger-Verlag, Berlin, 1987.

[8] B. R. Barmish, M. Corless and G. Leitmann, "A new class of stabilizing controllers for uncertain dynamic systems," *SIAM J. Control & Optim., vol 21, pp. 246-255, 1983.*

[9] I. R. Peterson, "Disturbance attenuation and H^∞ optimization : a design method based on the algebraic Riccati equation," *IEEE Trans. Autom. Cont., vol. 32, pp. 427-429, 1987.*

[10] M. A. Dahleh and J. B. Pearson, "l^1-optimal feedback for MIMO discrete-time systems," *IEEE Trans. Autom. Cont., vol. 32, pp. 314-322, 1987.*

[11] E. Yaz and A. Azemi, "Lyapunov-based nonlinear observer design for stochastic systems," *Proc. 29th CDC, vol. 1, pp. 218-219, 1990.*

A Model For Feedback Neural Nets With Supervised Learning

Jiansheng Hou and Fathi M. A. Salam
System and Circuits & Artificial Neural Nets Laboratories
Department of Electrical Engineering
Michigan State University
East Lansing, MI48824

Abstract: We present a new feedback network model, with weight updating, that specifies saddle points of the system and thus controls the regions of attractions of stable system equilibria. This models can in principle specify data stored and the size of the region associated with these data.

1. The Mathematical Model

To assure that we have a gradient system, which has a lot of desirable features, we start with the "energy" function:

$$E = -\frac{1}{2}\sum_i^n \sum_j^n w_{ij} \prod_{l=1}^m (v_i v_j - v_i^{*l} v_j^{*l}) - \sum_{i=1}^n I_i v_i + \sum_{i=1}^n \rho_i \int_0^{v_i} s_1^{-1}(\xi)d\xi + \sum_i^n \sum_j^n \alpha_{ij} \int_0^{w_{ij}} s_2^{-1}(\xi)d\xi,$$

where α_i, I_i, ρ_i are coefficients, $v_i = s_1(u_i)$, $w_{ij} = s_2(t_{ij})$, and s_1 s_2 are sigmoid functions (bounded, differentiable, monotone increasing diffeomorphism). Particularly, in our simulations, we used $s_1(u_i) = \tanh(u_i)$ and $s_2(t_{ij}) = 2\tanh(t_{ij})$. v^{*l} are desired patterns with $1 \leq l \leq m$.

$$c_i \dot{u}_i = -\frac{\partial E}{\partial v_i} \quad c_{ij} \dot{t}_{ij} = -\frac{\partial E}{\partial w_{ij}}$$

where c_i, and c_{ij} are coefficients to be chosen, and assuming $w_{ij} = w_{ji}$. Then, the general system differential equations and updating rule for a n-neuron m-pattern system are as follows:

$$\begin{cases} c_i \dot{u}_i = \sum_j^n w_{ij} v_j \left[\sum_{l=1}^m \prod_{\substack{k=1 \\ k \neq l}}^m (v_i v_j - v_i^{*^k} v_j^{*^k}) \right] - \rho_i u_i + I_i & i = 1, 2, \ldots n \\ c_{ij} \dot{t}_{ij} = \prod_{k=1}^m (v_i v_j - v_i^{*^k} v_j^{*^k}) - \alpha_{ij} t_{ij} & i, j = 1, 2, \ldots n \end{cases}$$

Observe that the derivative of the energy function is

$$\begin{aligned} \frac{dE}{dt} &= \sum_i^n \frac{\partial E}{\partial v_i}\frac{\partial v_i}{\partial u_i}\dot{u}_i + \sum_i^n \sum_j^n \frac{\partial E}{\partial w_{ij}}\frac{\partial w_{ij}}{\partial t_{ij}} t_{ij} \\ &= -\sum_i^n (\frac{\partial E}{\partial v_i})^2 (\frac{\partial v_i}{\partial u_i}) - \sum_i^n \sum_j^n (\frac{\partial E}{\partial w_{ij}})^2 (\frac{\partial w_{ij}}{\partial t_{ij}}) \end{aligned}$$

And thus, since $\frac{\partial v_i}{\partial u_i} > 0$, and $\frac{\partial w_{ij}}{\partial t_{ij}} > 0$, This gives $\frac{dE}{dt} \leq 0$. Consequently,

$$\dot{E} = 0 \iff \left\{ \frac{\partial E}{\partial v_i} = 0 \text{ and } \frac{\partial E}{\partial w_{ij}} = 0 \right\} \iff \left\{ \dot{u}_i = 0,\ \dot{t}_{ij} = 0 \right\}$$

In addition, the energy function E is bounded below and thus starting from any initial point, the system will move toward the direction where the total energy will be minimized and stop, when $\dot{E} = 0$, and E is finite, at an equilibrium.

2. Prototype Systems: Case Study

To understand the dynamics of such a system, we start with the simplest case.

2.1 One-neuron-one-pattern case

For one-neuron-one-pattern, the system dynamic equations are

$$\begin{cases} \dot{u} = wv - \rho u \\ \dot{t} = (v^2 - v^{*2}) - \alpha t \end{cases}$$

Here we have chosen $c_i = c_{ij} = 1$, $I_i = 0$, for simplicity. The system equilibria are the intersections of the following two functions:

$$
\begin{aligned}
f_1(u) &= \operatorname{atanh}\left(\frac{\rho u}{2v}\right) \\
f_2(u) &= \frac{v^2 - v^{*2}}{\alpha}
\end{aligned}
$$

Figure 1 and Figure 2 show the plot of the functions f_1 and f_2 for different coefficients α and ρ. It can be seen that the system may have one, three, or more equilibria (up to seven).

3. Conclusion

Over all, this proposed method has the following advantages over the learning algorithms proposed so far for feedback neural networks.

1. This model is a gradient system which is guaranteed to converge to equilibria and has no oscillation.
2. This method allows us to specify multiple equilibria of the system, (the specified values are near the values of system's equilibria).
3. The equilibria are saddle points. Thus set to be we can control the regions of attractions of the stable equilibria in the system. This is a desirable feature which can be used in categorization and grouping.

Using this new method, one can build multiple stable state one-neuron networks, which could be used as associative memory to store multiple data. Our concrete results are limited to one neuron cases. Future research will focus on multiple neuron cases and on specifying stable equilibria.

Acknowledgment:

This work is supported in part by ONR Grant N00014-89-J-1833, the Michigan Research Excellence Fund (REF), and NSF Grant ECS-8814027.

References

[1] F. M. A. Salam and Shi Bai, "A New Feedback Neural Network With Supervised Learning", *IEEE Trans. on Neural Networks*, Vol.2 No. 1 pp.170-173, January 1991.

[2] F. M. A. Salam, Yiwen Wang, and Myung-Ryul Choi, "On the Analysis of Dynamic Feedback Neural Nets ", *IEEE Trans. on Circuits and Systems* Vol. 38, No. 2. pp. 196-200, February 1991

Figures

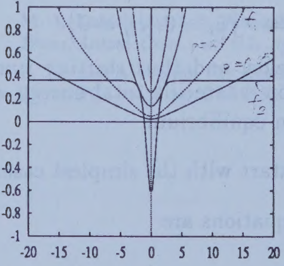


Figure 1

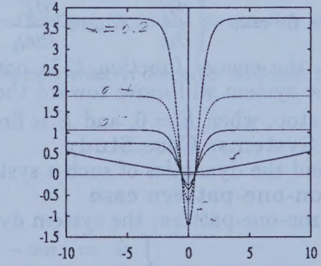


Figure 2

FEEDFORWARD ARTIFICIAL NEURAL NETWORK STRUCTURE FOR CHARACTER RECOGNITION

Maqbool Ahmad and Fathi M. A. Salam
System and Circuits & Artificial Neural Nets Laboratories
Department of Electrical Engineering
Michigan State University
East Lansing, MI 48824

Abstract

We decompose the character recognition problem into two parts: (1) feature extraction and (2) classification. The two networks are trained for the Arabic numerals using the gradient descent update law. The feature extraction is trained on a large data-base whereas the classification is trained only on new additional data. We show that this scheme considerably improves the character recognition capability of the network.

Summary

We have collected 100 different fonts including roman, bold, and italic versions of different classes from printed material [1]. Each of these printed fonts was scanned using the SC-7500 Toshiba scanner and was converted to the "Hips" format. The pixel images of these fonts were stretched appropriately to fit within a window of 17×15 pixel array while leaving a small margin on all sides. The gray levels of these pixels were set to be 0 or 1.

The network consists of the feature extraction and the classification networks, concatenated in a feedforward manner. The feature extraction network is designed to extract features locally using local connections to a previous layer, while the classification is done with global (i.e. full) connections. The output of each neuron is in the range (0,1) as the activation output is determined by the so called logistic function [2].

A partially connected feedforward artificial neural network (FFANN) with one hidden layer is chosen to extract features of input patterns. The input layer of this feature extraction network is a 17×15 array of nodes corresponding to these patterns which are the characters 0 through 9 of different fonts.

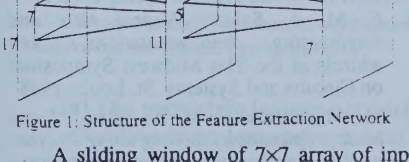


Figure 1: Structure of the Feature Extraction Network

A sliding window of 7×7 array of input nodes are connected to a neuron in a hidden layer. The window slides all over the input array moving one column or one row of 7-input nodes at a time. As a result the array-size of the hidden layer is 11×9. A similar sliding window of 5×5 array of the hidden layer neurons is connected to a neuron in the output layer. The resulting array-size of the output layer consequently is 7×5. The structure of the network is depicted in Figure 1.

Ten 7×5 array targets are constructed one for each class. The targets are constructed so as to retain the general shape of the characters and, in addition, be pseudo-orthogonal to each other. The targets are considered orthogonal if the hamming distance between them is equal to the dimension of the targets, namely 7×5=35. In the present case, the minimum and the maximum hamming distances between the targets we have constructed is 17 and 22, respectively. As a consequence, this coding has an error correction capability of up to 8 errors in the output pixels.

The capability of the feature extraction network can be measured by noting the correlation of the output digits with the targets. If the output is within a hamming distance of 8 of a target, it is classified to that target; otherwise we consider the network to be unable to classify the particular input.

The classification network has only one layer consisting of 10 neurons corresponding to the classes of characters, namely 0,1,...,9. The output of the feature extraction network from its 35 output nodes is directly fed into all of the classification neurons. The targets for the 10 different classes are chosen so that, for any particular class, the output for the corresponding node is high and all others are low. If the output of more than one node goes high, the network is considered unable to classify.

We used the gradient descent method with the error back-propagation learning technique for the training of the network. We trained the feature extraction and the classification networks separately with a view to achieve the following advantages:

(1) The two networks can be viewed as two separate modules,

(2) The feature extraction can be fully trained on a large data-base off-line and the classification network can be quickly trained (on-line) when new fonts are added to the data-base.

Following the presentation of an input/output pattern $p=1,2,...,P$, the rule for updating the weights w_{kj}, connecting the jth node in a layer to the kth node in the subsequent layer, is given by [3]

$$\dot{w}_{kj} = -\gamma \frac{\partial E}{\partial w_{kj}} \qquad (1)$$

Here γ is the learning rate and is chosen to be 1 and E is the Gaussian energy given as

$$E = \frac{1}{2}\sum_{p}\sum_{n} (t_{pn} - y_{pn})^2, \qquad (2)$$

where t_{pn} and y_{pn} are, respectively, the nth component of the desired and the actual output vectors for the input/output pattern p. The target vector t_p is formed by concatenation of the rows of the target matrix for pattern p.

For the integration of the update law (1), the simulator uses fourth order Runge-Kutta routine. The weights were initialized randomly by a random number generator. The value of any initial weight w is such that $-0.9 \le w \le 0.9$. A stopping criterion s was set to determine if the system has converged to a solution. The stoppage criterion s is defined as

$$s := \sum_{w} |\Delta w| \le 0.15, \qquad (3)$$

where w are all the weights in the network and

$$\Delta w = w(t + \Delta t) - w(t). \qquad (4)$$

The term Δt is the time step of the integration which is chosen to be 0.0005 secs. We have also used an *error criterion* to characterize the training: the lower the value of the error, the better is the training. The error parameter is chosen to be 400 and is defined as

$$\xi = \sum_{p}\sum_{n} (t_{pn} - y_{pn})^2. \qquad (5)$$

Half of the data-base was used as a training set. We trained the feature extraction network first. As the training of the network takes a long time we resorted to using an appropriate energy function E in the update law (1). We have used the "Polynomial" energy function E^p [4], given as

$$E^p = \frac{1}{2}\sum_{p}\sum_{n} (|t_{pn} - y_{pn}| + \frac{1}{2})^2, \qquad (6)$$

and the Exponential energy function F [5], given as

$$F = \kappa^{\mu E}. \qquad (7)$$

Here κ and μ are any positive constants and E could be the Gaussian (2) or the Polynomial energy function (6). We used $\kappa = e$ and $\mu = 0.004$ as long as $\partial \kappa^{\mu E} / \partial E \ge 1$.

For the Gaussian and the Polynomial energy, learning was terminated when the stoppage criterion s was achieved. On the other hand for the exponential energy, learning was stopped when the error criterion was achieved.

Table 1

	Gaussian	Polynomial	Exp Gaussian
ξ	526.45	638.96	399.96
steps	16205	18442	10598
correct	92.6%	93%	92.4%
incorrect	3%	1.8%	2.6%
reject	4.4%	5.2%	5%

When the network converged (to a local minimum), it could completely recognize the fonts in the training set, irrespective of the training method. The error(s) and the number of steps at convergence and the recognition performance on the test set is tabulated in Table 1.

The classification network was trained with the Gaussian energy function using the output of the feature extraction network as its input. The training set 1 and 2 consisted of the outputs when the training and test sets were input to the trained feature extraction network. On the termination of the training the network could completely recognize the fonts for which it was trained. The test results for the other fonts are given in Table 2.

Table 2

	Training set 1	Training set 2
correct	94%	99.8%
incorrect	2%	0%
reject	4%	0.2%

Note that when the feature extraction network is trained with the training set and the classification network is trained with the training set 2 (which is the output of the feature extraction network when the input is the test set), the character recognition improves considerably. The test set is actually the data which the user is adding for his work. Besides improving the recognition performance, this modular approach provides training convenience to users.

Acknowledgment: This work is supported in part by ONR Grant N00014-89-J-1833, the Michigan Research Excellence Fund (REF), and NSF Grant ECS-8814027.

References

[1]. W. J. Jespert, W. T. Berry and A. F. Johnson, The Encyclopedia of Type Faces, Blandford Press Ltd. London, 1970.

[2]. D. E. Rummelhart, G. E. Hinton and R. J. Williams, *Learning internal representations by error propagation,* in Parallel Distributed Processing: Explorations in the microstructures of cognition, Cambridge, MA: MIT Press, vol. 1, pp. 318-362, 1986.

[3]. F. M. A. Salam, *Neural Nets and Engineering Implementations,* key address at the 31st Midwest Symposium on Circuits and Systems, St. Louis, 1988.

[4]. M. Ahmad and F. M. A. Salam, *Error back propagation learning using Polynomial energy function,* under preparation.

[5]. M. Ahmad and F. M. A. Salam, *Error back propagation learning using Exponential energy function,* under preparation.

HANDWRITTEN NUMERIC CHARACTER RECOGNITION USING A MULTILAYER NEURAL NETWORK

Saida Benromdhane, Maqbool Ahmed and Fathi. M. A. Salam
Systems and Circuits & Artificial Neural Nets Laboratories
Department of Electrical Engineering
Michigan State University
East Lansing, MI 48824

Abstract

A recognition system for identifying handwritten digits is developed using a specific neural network structure. Layers of feedforward networks preprocess the digits by extracting features that enhance their dissimilarity.The features are defined based on an idealized geometry of the image/digit and/or they may approximate well known quantities such as moments and moments invariants. The advantage of the latter features is that they are invariant to translation, rotation and scaling.The collective tasks of the feedforward network implicitly integrate feature extraction with classification capabilities.

Introduction

The feature extraction task is handled using locally connected neural networks; however, the classification task is executed using a fully connected neural network. We have collected ten samples from ten different persons. The samples collected are as unique and different as possible in order to make the training set as representative as possible. When the training set is collected, two test sets are also collected. The first set is from the same ten persons and the second set is from ten other persons. Each of the collected handwritten numerals is scanned using the *SC 7500 Toshiba scanner* and stored eventually as a binary image. The images are then scaled so that each character fills a 17x15 frame while leaving a small margin on all sides.

The Feature Extraction Networks

We specialized two feedforward neural networks for the feature extraction task.

(a) The geometric feature network

One three-layered network uses local connectivity to map each handwritten digit to a prototype version of the same digit. Consequently all the training data of a digit, say 6, will be mapped to a prototype digit 6. In this process, the variations of the digit 6 is uniformly processed to the prototype digit 6. The input layer to this network is a 17x15 array corresponding to the input patterns. These patterns are the handwritten ten numerals 0 through 9. The input nodes accept the patterns of the input image and pass them to some of the neurons in the hidden layer. The array size of the hidden layer is 11x9. Each neuron of the hidden layer has an output that lies between 0 and 1, depending on the activation output of the logistic function [1]. The array size of the output is 7x5. In selecting a 7x5 array target corresponding to each input numeral, we sought to retain the general shape of the characters while being pseudo-orthogonal to each other. The orthogonality (of the targets) is determined by the Hamming distance among the targets which is 35.

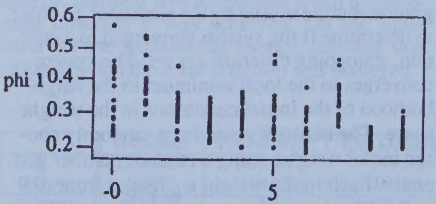
We remark that in this case it was not possible to retain the shape and retain orthogonality among the targets simultaneously. The maximum Hamming distance between the targets we have constructed is 22 while the minimum is 17. Consequently, the coding has an error correction capability of approximately eight errors (17/2) in the output array. The classification capability of the neural network is as follows: If the output is within a Hamming distance of eight of a target, it is classified to that target; otherwise, a reject decision is made and the network is unable to classify.

In the training process, we use the gradient descent algorithm with the continuous-time error update learning rule [2-3]. We have used a training simulator, developed in [2-3], incorporating the weight update law in C-language using a Sun Sparc station. In the integration of the update law, a fourth order Runge-Kutta integration routine is used by the simulator. In order to determine if the system converged to a solution, a stopping criterion s is set. The system converges to the local minimum in the neighborhood of the initial conditions in the weight space. The network starts from randomly chosen initial weights using a random number generator. Each initial weight w_{ij} ranges from -0.9 to 0.9. The parameters converge when a stoppage criterion s is achieved. An output error criterion is also set to stop the training. This error criterion is chosen sufficiently small.

After perfect training is achieved off-line, we begin testing. At this stage the simulator fixes the weights and performs a computation of an output from the preprocessed input numerals.This computation is done without any update of the weights. The test simulator compares the output computed with the targets to give one of the three results: Recognize correctly, unable to recognize, recognize incorrectly by substitution.

(b) The moments feature network:

The moments and the moments invariants may be viewed as features of an image. [4] A characteristic of these moments invariants is that they are invariant to (an image's) translation, rotation, and scaling. For each numeral 0-9 of the training set (20 samples), we computed a modified form of the seven moment-invariants.Thus, for each of the seven moment-invariants there are 20 values. For each numeral, we then computed the average value for each one of the seven moment invariants. And finally, we computed, for each moment, a global average for all values of the moment-invariants. To obtain a unique target for each numeral 0-9,we binarized the moment-invariants as follows: for each numeral, the local average of each moment-invariant is compared with the global average. We set the (binarized) values to 1 (respectively 0) if the local average is greater than (respectively lower than) the global average. It is essential to make sure that the targets are distinct. Thus we obtain a single target for the feature extraction neural network corresponding to each numeral. Now a 3-layered neural network can be trained to map each numeral of the training set to one of ten targets. It is anticipated that the trained neural network would approximate the functions of the moment-invariants. Consequently, the neural network may be viewed as a crude approximator for computing the moment invariants. It should be noticed, however, that the purpose of the neural network is not to approximate the exact expression of the moment-invariant functions but rather the invariance feature.



The Classifier

After training and testing the feature extraction network, we train the classification network. This network consists only of one layer. Its input is the output of the feature extraction network.The output has ten neurons (classes). This classification network refines the performance of the identification of the handwritten numerals.

Results

Table 1:

	Test set 1	Test set 2
correct	67	50
Incorrect	12	20
Reject	21	30

Table 2:

	Test set 1	Test set 2
correct	50	40
Incorrect	24	35
Reject	26	25

Table 3:

	Test set 1	Test set 2
correct	76	62
Incorrect	7	16
Reject	17	22

Acknowledgment

This work is supported in part By ONR grant N00014-89-J1833, The Michigan Research Excellence fund (REF) and NSF grant ECS-8814027.

References

[1] D. E. Rummelhart, G. E. Hinton and R.J. Williams, *Learning internal representation by error propagation in parallel distributed processing: Explorations in the microstructures of cognition, Cambridge*, MA: MIT Press, vol. 1, pages 318-362, 1986.

[2] F. M. A. Salam, *Neural Networks and engineering implementations*, Key address at the 31st Midwest Symposium on Circuits and Systems, St.Louis MO Aug 10-12 1988.

[3] F. M. A. Salam, *Artificial Neural Networks: basic theory and engineering implementation*, Department of Electrical Engineering, Michigan state university, East Lansing, MI 48823, October 1989.

[4] Anil K. Jain, *Fundanentals of digital image processing*, Prentice Hill information and system science series, Thomas kailath, series editor.

Neural Nets with Prespecified Memories

Jorge Aravena, Manju Hegde, Morteza Naraghi-Pour
Department of Electrical and Computer Engineering
Louisiana State University, Baton Rouge, LA 70803-5901

Summary. This paper presents a universal interpolation property for neural nets with high order synaptic connections. If the net is a Hopfield-type associative net, the user can specify an *arbitrary* set of memories and points in the attraction basin. If the net is of a feedforward type then the user can specify an arbitrary collection of input-output pairs and design a net that will provide *perfect interpolation* for all the pairs. The issue brought up by these nets is that of complexity. It is seen that in order to obtain these results one must increase the order of the synaptic connections for associative nets, or the number of hidden neurons for feedforward networks. The paper examines in detail the case of associative networks and proposes a tradeoff between complexity and capacity.

Basic Result. The mathematical result supporting the developments is derived from properties of the tensor product of vectors. Consider a given $n-$dimensional linear vector space, U. The symbol $x \otimes y$ will denote the tensor product of two vectors in a suitably defined tensor product space. The following function is derived from the study of polynomic systems [1].

$$\tau_p(x) = col\{x,\ x \otimes x,\ ...,\ \otimes^{(p)} x\};\ \ x \in U.$$

The expressions of τ_p for various values of p can be computed recursively using the formula

$$\tau_1(x) = x,$$
$$\tau_{p+1}(x) = col\{x, x \otimes \tau_p(x)\};\ \ p = 1, \ldots$$

The result which motivates the neural implementation shown here is stated below

Theorem 1 : If $\{x_1, ..., x_p\}$ is an arbitrary collection of distinct, non zero, vectors then the collection $\{\tau_p(x_1), ..., \tau_p(x_p)\}$ is always linearly independent.

REMARK: The result in the theorem gives only an upper bound in the complexity of the function, $\tau_k(x)$. This upper bound is reached if the starting vectors $\{x_1, \ . \ . \ . \ , x_p\}$ are all colineal. In most practical applications one would have linear independence for a collection of vectors $\{\tau_k(x_i),\ i = 1, .., p$ $k < p\}$. For notational convenience this paper will use the result stated in theorem 1.

The Neural Net For simplicity in the development all vectors take values in the set $\{1, -1\}$. Let $\{x_1, ..., x_p\}$ be desired collection of memories, let $\{\tau_p(x_1), ..., \tau_p(x_p)\}$ be the collection of linearly independent *lifted vectors* and let $G_p = [< \tau_p(x_i), \tau_p(x_j) >]$ be their non singular Grammian matrix. Define now the matrix , Y, with columns $\{y_1, \ . \ . \ . \ , y_p\}$, and the matrix T_p with columns $\{\tau_p(x_1), \ . \ . \ . \ , \tau_p(x_p)\}$. Let ξ be the state vector of the net, with each component being the output of the corresponding neuron. The procedure to update the net can is given by the expression

$$\mathcal{N}(\xi) = sgn\ \{Y\ G_p{}^{-1} T_p{}^{*} \tau_p(\xi)\ \}.$$

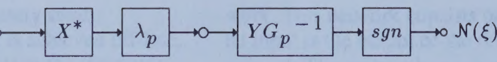
Using this rule one can establish the following result:

$$\boxed{y_k = \mathcal{N}(x_k),\ \ k = 1, \ . \ . \ . \ , p.}$$

Introducing the *lifting function,* $\lambda_p(d) = d + d^2 + d^3 + \ . \ . \ . \ + d^p$, with the convention that for matrices the notation $\lambda_p(A)$ indicates componentwise evaluation, one can prove that

$$T_p{}^{*} \tau_p(\xi) = \lambda_p(X^{*} \xi)$$

This result can be interpreted as a fully interconnected feedforward net of the form shown below.



This net has an input layer with size equal to the dimension of the input vector, the output layer has the size of the output vector. However, the hidden layer has the dimension equal to the *number of vectors in the training set.* Thus the tradeoff of complexity for perfect learning is apparent.

The expression for $\mathcal{N}(\xi)$ can also be viewed as a synchronous updating technique for an associative net. The corresponding componentwise updating is easily derived. For the $i-th$ neuron one can write

$$[\mathcal{N}(\xi)]_i = sgn[\sum_j \alpha_{ij}\lambda_p(x_j{}^*\xi)] = sgn[\sum_k[\sum_j \alpha_{ij}\tau_{jk}]\xi_k + \sum_{k,l}[\sum_j \alpha_{ij}\sigma_{jkl}]\xi_k\xi_l + \ldots]$$

The number of maximum order of the connections is influenced by the memories that must be stored; Theorem 1 provides an upper bound for this number. In this case it is clear that an arbitrary number of inputs can be mapped to a single output vector and will become part of the attraction basin. Again one can see the tradeoff of complexity and capacity.

One can set a limit to the order of the connections and require the stored memories to have a stable basin. This approach controls the complexity of the net but also limits capacity. In this regard one can prove:

Lemma 1: For the net with updating equation $[\mathcal{N}(\xi)]_i = sgn[2\sum_k T_{ik}{}^{(2)}\xi_k + 3\sum_{k,l} T_{ikl}{}^{(3)}\xi_k\xi_l + \ldots]$. *If the synaptic weights,* $T_{ik}{}^{(2)}$, $T_{ikl}{}^{(3)}$, *..., are invariant under any permutation of the subindices, and they are zero whenever two subindices are equal, then any asynchronous updating converges to a stable memory.*

In order to quantify the increase in capacity for the constrained high order net, several experiments with randomly selected memories have been simulated. The table below give the number of failures of a net with ***N*** neurons to store ***p*** patterns in 100 trials. The weights have been determined using an adaptive relaxation iterative procedure [2]. This procedure has geometric convergence rate and will reach the solution (if it exists) in a finite number of steps. The top number corresponds to a conventional Hopfield net and the lower to a net with third order synaptic connections.

$p\rightarrow$	15	16	17	18	. . . 20	. . .30	
$N\downarrow$	39	74	81	....			*Hopfield*
15	4	6	5	10	12		*3rd order*
	33	38	65	....			*Hopfield*
16	0	0	0		0	3	*3rd order*
	0	0	0	4	52	. . .	*Hopfiled*
20	0	0	0	0	0	0	*3rd order*

The better performance of the third order net is apparent. However, the number of the synaptic weights and the time required to train the net increase rapidly. Further testing is still in progress. Results tend to indicate that the capacity increases very rapidly with both the number of neurons, ***N***, and particularly with the maximum order of the synaptic connections.

Selected References:[1] Porter W. A. *Polyarrays as distributive memory arrays*, **Proc. CommCon 88**, pp 723-734, Baton Rouge, Oct 1988.

[2] Jiang, X. C., M. Hegde, M. Naraghi-Pour, *Neural Network design using linear programming and relaxation,* **Proc. 1990 IEEE ISCAS**, pp 1090-1093, New Orleans, 1990.

LIMITATIONS OF CONSTRAINED CONNECTIVITY NEURAL NETWORKS

Manju Hegde, Mort Naraghi-Pour and G. Nandakumar
Department of Electrical and Computer Engineering
Louisiana State University
Baton Rogue, LA 70803-5901

1. Regular Networks with Constant Degree

A nonzero off-diagonal element T_{ij} in the connectivity matrix T represents the presence of a connection between neurons i and j. On the other hand, a zero off-diagonal element T_{ij} corresponds to the lack of a connection between neurons i and j. Therefore, restricted connectivity/topology imposes a specific pattern of zeros in the connectivity matrix. The adaptive relaxation rules introduced in [1] are capable of designing networks with prespecified network topologies. It is intuitively plausible that as the connectivity of the network is reduced, the capacity is reduced as the number of variables which can be selected to satisfy the memory requirements is reduced.

In Theorem 1 we derive lower and upper bounds on the probability of failure to store 2 randomly chosen memories in a network with some specified topology. Let $\mathbf{V}^1$ and $\mathbf{V}^2$ denote the 2 random memories, $\mathbf{V}^s = (V_1^s, \ldots, V_N^s)$, $s = 1,2$ with $V_j^s \in \{-1,1\}$. The V_j^s's are chosen independently with equal probability of being +1 or -1. We say the event, "failure to store the 2 memories," F_2^N, occurs if there does not exist a zero diagonal symmetric matrix which can store both memories simultaneously. Now we obtain Theorem 1. (The proof as well as more extensive simulation results may be found in [2].)

Theorem 1: Consider the graph of a neural network with N neurons and a constant degree for every node. Let $d(N)$ denote the degree of each node and let $C(N)$ denote the cardinality of the largest subset of nodes of the graph which has minimum graph distance 3. Then

$$1 - \left(1 - \left(\frac{1}{2}\right)^{d(N)}\right)^{C(N)} \leq P(F_2^N) \leq N\left(\frac{1}{2}\right)^{d(N)}.$$

Corollary 1: The asymptotic capacity of the family of regular graphs with $d(N) = d$, is 1.

In Tables 1 and 2 we provide simulation results for the cases of grids and double-rings which demonstrate their worsening ability to store 2 random memories. Corollary 1 tells us that although from implementation considerations, it would be desirable to have a topology with constant degree (independent of N), it is *necessary* for the node degree of the topology to increase with N at least logarithmically if the topology is to be useful as an associative memory. We are therefore led to investigate regular graphs whose node degree $d(N)$ increases with N, the number of nodes (neurons).

2. Regular Networks with Increasing Node Degrees

In [2] we show the following for hypercube graphs ($d(N)=\log_2(N)$): for any $\varepsilon>0$ for sufficiently large N,

$$P(F_N^2) \geq \frac{1}{2} - \varepsilon.$$

Thus we see that the capacity of a hypercube graph is also 1. The simulations presented in Table 3 confirm this.

The square-root N family of graphs is defined for $N = 4n^2$, $n = 1,2,3,\ldots$ as follows. The set of nodes of this graph is $\{1,2,\ldots N\}$. Each node i is connected to nodes $i+1, i+2, \ldots, i+\frac{\sqrt{N}}{2}, i-1, i-2, \ldots, i-\frac{\sqrt{N}}{2}$ (where the additions and subtractions are modulo N). For this family of graphs, $d(N) = \sqrt{N}$.

The $p \times q$ fully connected trellis can be defined as follows. It has pq nodes which may be denoted as $\{a_{ij}, i=1,\ldots,p, j=1,\ldots,q\}$. Let $m = (i+1) \bmod q$. Each a_{ik} is connected to a_{mn} for all n. For this family of graphs, $d(N) = 2(\frac{N}{q})$.

Theorem 1 does not rule out non-trivial capacities for such networks. In Table 4 we find that the capacity of the square-root N family for $N = 144$ is small (approximately 3). In Table 5 we find the capacity of a trellis network with

---

This work was supported by the Louisiana Education Quality Support Fund through Contract No. LEQSF (1989-91)-RD-A-5.

48 neurons. Finally, in Table 7 we present the corresponding results for fully connected networks for which $d(N) = N - 1$. As expected, these topologies have larger capacities than any of the other topologies.

References

[1] M. Hegde, M. Naraghi-Pour, G. Nandakumar, "Flexible learning rules for neural networks," submitted to *Neural Networks*, June 1991.

[2] M. Hegde, M. Naraghi-Pour, G. Nandakumar,"Constrained Connectivity and Capacity of Neural Networks", submitted to *IEEE Transactions on Neural Networks.*

Table 1: No. of failures in 100 trials to store 2 memories with p x q Grids

p q	2	3	4	5	6
2	64	61	74	70	69
4	70	83	75	81	91
6	80	81	86	86	94
8	78	91	98	92	98

Table 2: No. of failures in 100 trials to store 2 memories with the Double Ring topology

No. of neurons	No. of failures
10	52
20	82
40	95

Table 3: No. of failures in 100 trials to store 2 memories with Hypercubes

# of neurons	2	4	8	16	32	64	128
# of failures	62	64	50	50	58	47	63

Table 4: No. of failures in 100 trials to store 2 memories with $\sqrt{N}$ Topology

No. of neurons	4	16	36	64	100	144
No. of failures	64	59	38	15	9	1

Table 5: Probabilities of failure to store indicated no. of patterns with 16x3 Trellises (48 Neurons)

No. of memories	29	30	31
Relaxation	0.027	0.073	0.184

Table 6: Probabilities of failure to store indicated no. of memories with
30 neuron fully connected network

No. of memories	29	30	31	32	33
Relaxation	0.027	0.044	0.059	0.137	0.303

ANALYSIS AND SYNTHESIS OF A CLASS OF DISCONTINUOUS DISCRETE-TIME NEURAL NETWORKS WITH INFINITE GAIN*

G. Yen A. N. Michel

Department of Electrical Engineering
University of Notre Dame
Notre Dame, Indiana 46556

ABSTRACT

In the present paper we investigate the qualitative properties of a class of discrete-time neural networks described by a system of first order ordinary difference equations with discontinuous right-hand side. The class of systems considered retains the basic structure of Hopfield type neural networks with the nonlinearities having infinite gain. For the present model we establish *qualitative results* which enable us to generalize the solutions of discrete-time systems in the discontinuous surfaces and to characterize the set of system equilibrium points (distribution of equilibria in the state space and stability properties of the equilibrium points). In addition, we develop an efficient *synthesis procedure* utilizing the *eigenstructure method*, for the present class of neural networks. The synthesized networks are capable of learning new patterns as well as forgetting learned patterns. The resulting neural networks can easily be implemented in digital hardware. Furthermore, when implemented by a serial processor, the present results offer extremely efficient means of simulating corresponding continuous-time neural networks (modelled by systems of ordinary differential equations). The applicability of the present results is demonstrated by means of several specific examples.

I. INTRODUCTION

We consider neural networks described by discontinuous difference equations of the form

$$x(k+1) = F(x(k)) = Ax(k) + TS(x(k)) + I, \qquad k=0,1,2,\ldots \qquad \text{(DN)}$$

where $x=(x_1,\ldots,x_n)^T \in R^n$, $F: R^n \rightarrow R^n$ is a measurable function, $A=\mathrm{diag}(-A_{ii})$ is an n×n constant real matrix with each $|A_{ii}|<1$, $1\leq i\leq n$, $T=[T_{ij}]$ is an n×n constant real matrix, $I=(I_1,\ldots,I_n)^T$ is a constant real vector, and $S: R^n \rightarrow R^n$ is a measurable function defined by $S(x)=(\mathrm{sgn}(x_1),\ldots,\mathrm{sgn}(x_n))^T$, where $\mathrm{sgn}(x_i)=1$, when $x_i>0$, $\mathrm{sgn}(x_i)=-1$, when $x_i<0$, and $\mathrm{sgn}(x_i)$ is undefined, when $x_i=0$.

Thus, F is not defined on M where $M=\{x\in R^n$: there is at least one i, $1\leq i\leq n$, such that $x_i=0\}$. The set M is said to make up the *discontinuous surfaces* of system (DN). The results which we establish for system (DN) fall into one of two categories. One type of result addresses the *stability analysis* of system (DN) while the other type pertains to *synthesis procedure* for system (DN).

II. ANALYSIS

Due to space limitation, we provide only a summary of qualitative results established in the present section.

a) Since the function F(·) is not continuous, a generalized definition of solutions for system (DN) is developed. On the surfaces M where F(·) is discontinuous, solutions in the *sliding mode* are considered [5]-[6]. For a given initial condition, the solutions of system (DN) exist for all $k\in Z^+$, where Z^+ denotes the set of non-negative integers.

---

*Supported in part by the National Science Foundation under grant ECS 88-02924.

b) The concept of *equilibrium points* for system (DN) is made precise. We show that in each of 2^n regions which are separated by M, there exists at most one asymptotically stable equilibrium point.
c) If in one of the above regions, say H, there is an equilibrium point, say x_e, then H is contained in the domain of attraction of x_e.
d) We call the elements in the range of the function S the *output vectors* of system (DN). We show that there is a one-to-one correspondence between *stable output vectors* and asymptotically stable equilibrium points in R^n-M.

Under the assumptions that T is symmetric and the main diagonal elements, T_{ii}, $1 \leq i \leq n$, of T are non-negative, we associate with system (DN) an energy function, and we show that:
e) The set of *local minimum points* of E is contained in the set of stable output vectors of system (DN). These two sets are equal if $T_{ii}=0$, $1 \leq i \leq n$.
f) Solutions of system (DN) will not be in the sliding mode.
g) If T is positive semidefinite, along each output sequence, the energy function decreases monotonically, and system (DN) will not exhibit periodic solutions.
h) If T is not positive semidefinite, g) is true with respect to *asynchronous solutions* of system (DN).
i) Utilizing the energy function approach, system (DN) is proved to be globally stable.

III. SYNTHESIS

The present section is divided into two parts. In Subsection A, we develop a synthesis procedure (*the eigenstructure method*) for system (DN) which is in the spirit of our earlier work in [3], [4]. In Subsection B, this synthesis procedure is further generalized to provide it with learning and forgetting capabilities.

A. A Synthesis Procedure for System (DN)

There are four terms in the right-hand side of system (DN): the diagonal matrix $A=\mathrm{diag}(-A_{ii})$, the nonlinear vector function $S(x)=(\mathrm{sgn}(x_1),\ldots,\mathrm{sgn}(x_n))^T$, the $n \times n$ matrix $T=[T_{ij}]$, and the vector $I=(I_1,\ldots,I_n)^T$. With the nonlinear function S(x) we fix some of the basic properties of system (DN). The values of A_{ii}, $1 \leq i \leq n$, in the matrix A do not affect the locations of the stable output vectors of the system (DN). By properly scaling each component of x, we may consider $A_{ii}=0$, $1 \leq i \leq n$. In this case, the system assumes the form

$$x(k+1) = TS(x(k)) + I, \qquad k=0,1,2,\ldots. \qquad \text{(DN)}$$

Accordingly, the basic synthesis problem which we will address may be stated as follows:
Synthesis Problem : Given m vectors in B^n ($B^n=\{x \in R^n: x_i=1 \text{ or } -1,\ 1 \leq i \leq n\}$), say $\alpha^1,\ldots,\alpha^m$, with the matrix A=0, and the nonlinear function $S(x)=(\mathrm{sgn}(x_1),\ldots,\mathrm{sgn}(x_n))^T$ fixed, how can we properly choose an *optimal* pair {T, I} such that the resulting system (DN) satisfies the properties enumerated below ?

1) $\alpha^1,\ldots,\alpha^m$ are stable output vectors of system (DN).
2) The system (DN) has no oscillatory output sequences.
3) The total number of the spurious stable output vectors (i.e., stable output vectors of system (DN) contained in the set $B^n - \{\alpha^1,\ldots,\alpha^m\}$) is as small as possible.
4) The domain of attraction of each α^i, $1 \leq i \leq m$, is as large as possible. ■

The analysis results developed in the analysis section enable us to to pursue the following approach:
Synthesis Strategy : Given m vectors in B^n, say $\alpha^1,\ldots,\alpha^m$, find an $n \times n$ matrix $T=[T_{ij}]$ and a vector $I=(I_1,\ldots,I_n)^T$ such that:

1) T is symmetric and $T_{ii} \geq 0$, $1 \leq i \leq n$.

2) $x_{\alpha^i}=\alpha^i$, where $\tau_1 x_{\alpha^i}=T\alpha^i+I$, and $\tau_1>1$.

3) Each α^i is a (local) minimum of the energy function E of the system (DN), where E: $B^n \to R$ is defined by $E(\alpha) = -1/2\alpha^T(T-E_n)\alpha - \alpha^T I$, where E_n denotes the n×n identity matrix.

4) Each main diagonal element of T is as small as possible. ■

Remark 3.1. In the following, we provide the rationale for the above synthesis strategy.

1) With T symmetric and $T_{ii} \geq 0$, $1 \leq i \leq n$, system (DN) will have no oscillatory output sequences.

2) For $1 \leq i \leq m$, $x_{\alpha^i}=\alpha^i$ implies α^i is a stable output vector for the designed system (DN). (In order for α^i to be a stable output vector, we actually require only that $S(x_{\alpha^i})=\alpha^i$. The stronger condition $x_{\alpha^i}=\alpha^i$ for $1 \leq i \leq m$ will be used in Theorem 3.2 below to determine spurious stable output vectors.)

3) We cannot specify precisely the domain of attraction of each α^i. The strategy which we use here is to place each α^i at the location which corresponds to a (local) minimum of the energy function for the synthesized system.

4) The output sequence of the synthesized system may get stuck at some vector which is not a minimum of the energy function when $T_{ii}>0$, $1 \leq i \leq n$. Therefore, smaller values for each T_{ii} will correspond to fewer spurious stable output vectors and this will correspond to larger domains of attraction for the desired stable output vectors. ■

Suppose that we are given m vectors in B^n, say $\alpha^1,\dots,\alpha^m$. Let

$$L = \mathrm{Span}(\alpha^1-\alpha^m,\dots,\alpha^{m-1}-\alpha^m), \qquad L_a = \mathrm{Aspan}(\alpha^1,\dots,\alpha^m). \tag{3.1}$$

Then L is the linear subspace of R^n spanned by the (m-1) vectors $\alpha^1-\alpha^m,\dots,\alpha^{m-1}-\alpha^m$, L_a denotes the affine subspace of R^n generated by the vectors $\alpha^1,\dots,\alpha^m$, and $L_a=L+\alpha^m$. Assume that k=dim(L), $\{u_1,\dots,u_k\}$ is an orthonormal basis of L, and $\{u_{k+1},\dots,u_n\}$ is an orthonormal basis of $L^\perp$. Let $U^+=[u_1,\dots,u_k]$, $U^-=[u_{k+1},\dots,u_n]$,

$$T^+ = \sum_{i=1}^{k} u_i u_i^T = U^+(U^+)^T, \qquad T^- = \sum_{i=k+1}^{n} u_i u_i^T = U^-(U^-)^T, \tag{3.2}$$

and let

$$T_\tau = \tau_1 T^+ - \tau_2 T^-, \qquad I_\tau = \tau_1\alpha^m - T_\tau \alpha^m, \tag{3.3}$$

where $\tau_1, \tau_2 \in R$ are parameters. For this class of synthesized systems

$$x(k+1) = T_\tau S(x(k)) + I_\tau, \qquad k=0,1,2,\dots \qquad (\mathrm{DN}_\tau)$$

we have the following results.

Lemma 3.1. 1) T^+ and T^- depend only on the given set $\{\alpha^1,\dots,\alpha^m\}$ and they are independent of the choice of the reference vector α^m and of the orthonormal basis $\{u_1,\dots,u_n\}$ of R^n. This is also true for T_τ and I_τ, for any $\tau_1, \tau_2 \in R$.

2) For any τ_1 and τ_2, T_τ is symmetric.

3) For any τ_1 and τ_2 and for any $\alpha \in L_a$, $T_\tau\alpha+I_\tau=\tau_1\alpha$. In particular, for each α^i, $1 \leq i \leq m$, $T_\tau\alpha^i+I_\tau=\tau_1\alpha^i$.

4) For $\tau_2'<\tau_2''$, we have $T_{\tau',ii}\geq T_{\tau'',ii}$, $1\leq i\leq n$ where $T_{\tau',ii}$ and $T_{\tau'',ii}$ are the *i*th diagonal elements of matrix T_{τ}', and matrix T_{τ}'' determined by the parameters τ_2' and τ_2'', respectively.

5) When $\tau_1=1$ and $\tau_2=-1$, T_τ is an n×n identity matrix and $I_\tau=0$.

6) T_τ is positive semidefinite if and only if $\tau_1\geq 0$, $\tau_2\leq 0$.

7) There is a real value denoted by $\tau_{2,op}=\min\{\tau_1 T^+_{ii}/T^-_{ii},\ 1\leq i\leq n\}$ such that for $\tau_1>1$ and $0\leq\tau_2\leq\tau_{2,op}$, $T_{\tau,ii}\geq 0$, $1\leq i\leq n$. If for any $\tau_1>1$, $\tau_2>\tau_{2,op}$, there will be an integer i such that $T_{\tau,ii}<0$.

8) For a given $\tau_1>1$, if $-1<\tau_2'<\tau_2''$ and if $\alpha\in B^n$ is a stable output vector of system (DN_τ'') with parameters τ_1 and τ_2'', then α is also a stable output vector of system (DN_τ') with parameters τ_1 and τ_2'. ■

For fixed $\tau_1>1$, $\tau_2>-1$, consider the energy function E: S→R,

$$E(x) = -\frac{1}{2}x^T(T_\tau - E_n)x - x^T I_\tau \tag{3.4}$$

where $S=\{x\in R^n: |x|^2=x^Tx=n\}$. We have that $B^n\subset S$ and the restriction of the function E on B^n is the associated energy function of (DN_τ). We have the following results.

Lemma 3.2. When $\tau_1>1$, $\tau_2>-1$, $x\in S$ is a local minimum of function E if and only if $x\in L_a$. Thus, when the function E is restricted to B^n, each vector in $B^n\cap L_a$ is a local minimum. ■

From Remark 3.1 and from Lemmas 3.1 and 3.2, the following result follows:

Theorem 3.1. 1) For any τ_1, $\tau_2\in R$, each $\alpha\in B^n\cap L_a$ is a stable output vector of system (DN_τ). In particular, this is true for $\alpha^1,\ldots,\alpha^m$.

2) For any τ_1, $\tau_2\in R$, T_τ is symmetric.

3) When $\tau_1>1$, $\tau_2>-1$, each $\alpha\in B^n\cap L_a$ is a local minimum of the energy function E. In particular, this is true for $\alpha^1,\ldots,\alpha^m$.

4) When $\tau_1>1$, $\tau_2>-1$, the value of each main diagonal element of T_τ and the number of spurious stable output vectors of system (DN_τ) decrease as τ_2 increases.

5) When $\tau_1>1$, $-1<\tau_2<\tau_{2,op}$, each $\alpha\in B^n\cap L_a$ behaves as an attractor. In particular, this is true for $\alpha^1,\ldots,\alpha^m$.

6) Taking $\tau_1>1$, $\tau_2=0$, we obtain the system

$$x(k+1) = T_0S(x(k)) + I_0, \tag{DN_0}$$

where

$$T_0=\tau_1T^+, \qquad I_0=\tau_1\alpha^m - T_0\alpha^m, \tag{3.5}$$

and T_0 is positive semidefinite. The system (DN_0) is *optimal* in the sense that it has the fewest number of spurious stable output vectors among the systems (DN_τ), $\tau_2=0$, having no oscillatory output sequences with respect to *all of their solutions*.

7) Taking $\tau_1>1$, $\tau_2=\tau_{2,op}$, we obtain the system

$$x(k+1) = T_{op}S(x(k)) + I_{op}, \tag{DN_{op}}$$

where

$$T_{op}=\tau_1T^+ - \tau_{2,op}T^-, \qquad I_{op}=\tau_1\alpha^m - T_{op}\alpha^m. \tag{3.6}$$

The system (DN_{op}) is *optimal* in the sense that it has the fewest number of spurious stable output vectors among systems (DN_τ), $\tau_2>-1$, having no oscillatory output sequences with respect to *all of their asynchronous solutions*. ■

If $\alpha\in B^n$ is a stable output vector of system (DN_τ) for $\tau_1>1$, $\tau_2>-1$, $x_{\alpha,\tau}$ is an asymptotically stable equilibrium point of system (DN_τ). Whenever an output sequence of system (DN_τ) converges to α, the corresponding solution of system (DN_τ) converges to $x_{\alpha,\tau}$. By Lemma 3.1, the final state vector of (DN_τ) is equal to the final output vector of (DN_τ) if and only if $\alpha\in L_a$, where $\tau_1x_{\alpha,\tau}=T_\tau\alpha+I_\tau$. If $\alpha\notin L_a$, then $|\tau_1x_{\alpha,\tau}|\to+\infty$, as $\tau_2\to+\infty$.

Theorem 3.2. For system (DN_τ), $\tau_1>1$, $-1<\tau_2\leq\tau_{2,op}$, any of the spurious stable output vectors which do not belong to L_a can be determined by comparing the final output vector α^∞ and final state vector x_{α^∞}. If $\alpha^\infty\neq x_{\alpha^\infty}$, α^∞ is not in the given data set $\{\alpha^1,...,\alpha^m\}$. ■

We summarize the above discussion in the synthesis procedure given below (called *the eigenstructure method*).

Synthesis Procedure : Suppose we are given m vectors $\Lambda=\{\alpha^1,...,\alpha^m\}$ in B^n, which are to be stored as stable output vectors in an n-dimensional system (DN). We proceed as follows:

1) Compute the $n\times(m-1)$ matrix $Y_m=[\alpha^1-\alpha^m,...,\alpha^{m-1}-\alpha^m]=[y_1,...,y_{m-1}]$, where $y_i=\alpha^i-\alpha^m$, $1\leq i\leq m-1$, and the subscript m for matrix Y_m denotes the number of stable output vectors to be stored in the system (DN).

2) Perform a singular value decomposition of Y_m and obtain the matrices U, V and Σ_m such that $Y_m=U\Sigma_mV^T$, where U and V are left and right singular matrices, respectively, and where Σ_m is a diagonal matrix with the singular values of Y_m on its diagonal. Then $U = [U^+_m \quad U^-_m]$, where $U^+_m=[u_1,...,u_k]$, $U^-_m=[u_{k+1},...,u_n]$, and $L_m=\mathrm{Span}(y_1,...,y_{m-1})$. From the properties of singular value decomposition we know that $k=\mathrm{rank}(\Sigma_m)$, $\{u_1,...,u_k\}$ is an orthonormal basis of L_m, and $\{u_{k+1},...,u_n\}$ is an orthonormal basis of $L^\perp_m$.

3) Compute the matrices

$$T^+_m = \sum_{i=1}^{k} u_i u_i^T = U^+_m(U^+_m)^T, \qquad T^-_m = \sum_{i=k+1}^{n} u_i u_i^T = U^-_m(U^-_m)^T.$$

4) Choose parameters τ_1, τ_2 and compute

$$T_\tau = \tau_1T^+_m - \tau_2T^-_m, \qquad I_\tau = \tau_1\alpha^m - T_\tau\,\alpha^m.$$

In particular, the optimal pair $\{T_{op}, I_{op}\}$ is given by (DN_{op}) with parameters τ_1, $\tau_2=\tau_{2,op}$, where $\tau_{2,op}=\min\{\tau_1T^+_{ii}/T^-_{ii},\ 1\leq i\leq n\}$. Then all vectors in $B^n\cap L_a$, including $\alpha^1,...,\alpha^m$ will be stored as stable output vectors in the system (DN_τ) ($L_a=\mathrm{Aspan}(\alpha^1,...,\alpha^m)$). ■

Remark 3.2. 1) Whether a vector $\alpha\in B^n$ is a stable output vector of system (DN_τ) can be checked by the following criterion: $\alpha\in B^n$ is a stable output vector of (DN_τ) if and only if $\min\{\alpha^i\cdot x_{\alpha,\tau,i}\text{: } 1\leq i\leq n\}\geq0$, where $\tau_1x_{\alpha,\tau}=T_\tau\alpha+I_\tau$, and $x_{\alpha,\tau,i}$ is the *i*th component of $x_{\alpha,\tau}$.

2) Whether a stable output vector of system (DN_τ) is in L_a or not can be checked by the following criterion: $\alpha\in L_a$ if and only if $\alpha=x_{\alpha,\tau}$ where $\tau_1x_{\alpha,\tau}=T_\tau\alpha+I_\tau$. ■

B. Learning and Forgetting Algorithm for System (DN)

In synthesizing artificial feedback neural networks, we usually assume that all desired vectors to be stored are known *a priori*. However, in real-world applications, this is usually not the case. Sometimes we are also required to update the stored vectors. In the present subsection we incorporate learning and forgetting capabilities into the eigenstructure method, where *learning* is defined as the ability of adding vectors to be stored as asymptotically stable equilibria to an existing set of stored vectors in a given network, and where *forgetting* is defined as the ability of deleting specified vectors from a given set of stored equilibria in a given network.

Now suppose that system (DN_τ) has been designed by the eigenstructure method in such a manner that every vector in $\Lambda=\{\alpha^1,...,\alpha^m\}$ is stored as a stable output vector and suppose we desire to store an additional vector (i.e., we wish to *learn* a new vector) α^{m+1} such that $\Lambda=\{\alpha^1,...,\alpha^m,\alpha^{m+1}\}$. From Lemma 3.1, the synthesized system (DN_τ) is independent of the choice of α^m as the reference vector to construct matrix Y_m. Let

$$y_m = \alpha^{m+1} - \alpha^m, \qquad Y_{m+1} = [Y_m, y_m]. \tag{3.7}$$

In accordance with the above synthesis procedure, we desire to find the left singular matrix U from the singular value decomposition of Y_{m+1}, such that U^+_{m+1} contains the orthonormal basis of L_{m+1}, and U^-_{m+1} contains the orthonormal basis of $L^\perp_{m+1}$, where $L_{m+1}=\mathrm{Span}(Y_m,y_m)$. Utilizing the Gram-Schmidt orthogonalization procedure on the vector y_m, we are able to check whether the set $\{Y_m, y_m\}$ spans a (k+1)-dimensional subspace of L_{m+1} or a k-dimensional subspace of L_{m+1}. We have

$$t_{k+1} = y_m - \sum_{j=1}^{k} (y_m^T u_j)\, u_j. \tag{3.8a}$$

If t_{k+1} is equal to zero, then y_m does not span a new dimension from L_m (i.e., $k=\mathrm{rank}(Y_{m+1})$). The left singular matrix $U=[u_1,...,u_n]$ obtained from the previous step still satisfies the requirements that $\{u_1,...,u_k\}$ is an orthonormal basis of L_{m+1} and $\{u_{k+1},...,u_n\}$ is an orthonormal basis of $L^\perp_{m+1}$. If t_{k+1} is not equal to zero, we let

$$\tilde{u}_{k+1} = \frac{t_{k+1}}{|t_{k+1}|}. \tag{3.8b}$$

Then $\{u_1,...,u_k,\tilde{u}_{k+1}\}$ constitutes an orthonormal basis of L_{m+1}. To find $U^-_{m+1}= [\tilde{u}_{k+2},...,\tilde{u}_n]$, such that $\{\tilde{u}_{k+2},...,\tilde{u}_n\}$ is an orthonormal basis of $L^\perp_{m+1}$, we construct the tentative vectors $t_{k+2},...,t_n \in R^n$ as follows:

$$\begin{aligned}
t_{k+2} &= [t_{k+2,1},\ t_{k+2,2},\ ...,\ t_{k+2,k},\ t_{k+2,k+1},\ 1,\ 0,...,\ 0]^T,\\
t_{k+3} &= [t_{k+3,1},\ t_{k+3,2},\ ...,\ t_{k+3,k},\ t_{k+3,k+1},\ 0,\ 1,...,\ 0]^T,\\
&\ \ ...\\
t_n &= [t_{n,1},\ \ t_{n,2},\ \ ...,\ t_{n,k},\ \ t_{n,k+1},\ \ 0,\ 0,...,\ 1]^T.
\end{aligned}$$

By construction, the vectors $t_{k+2},...,t_n$ are linearly independent. We now need to ensure that the vectors $t_{k+2},...,t_n$ are orthogonal to the vectors $u_1,...,u_k$, and $\tilde{u}_{k+1}$ by solving the unique solution of the equation

$$\begin{bmatrix} u_{1,1} & u_{1,2} & \cdots & u_{1,k} & u_{1,k+1} \\ u_{2,1} & u_{2,2} & \cdots & u_{2,k} & u_{2,k+1} \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ u_{k,1} & u_{k,2} & \cdots & u_{k,k} & u_{k,k+1} \\ \tilde{u}_{k+1,1} & \tilde{u}_{k+1,2} & \cdots & \tilde{u}_{k+1,k} & \tilde{u}_{k+1,k+1} \end{bmatrix} \begin{bmatrix} t_{k+i,1} \\ t_{k+i,2} \\ \cdot \\ \cdot \\ \cdot \\ t_{k+i,k} \\ t_{k+i,k+1} \end{bmatrix} = \begin{bmatrix} -u_{1,k+i} \\ -u_{2,k+i} \\ \cdot \\ \cdot \\ \cdot \\ -u_{k,k+i} \\ -\tilde{u}_{k+1,k+i} \end{bmatrix}, \qquad i=2,\dots,n-k. \tag{3.9}$$

The matrix inversion for (3.9) always exists since the rank(U^+_{m+1})=k+1, and since the first k+1 columns of U^+_{m+1} can be rearranged to ensure a nonsingular matrix, if necessary. Since the vectors $t_{k+2},\dots,t_n$ are independent and orthogonal to the basis $\{u_1,\dots,u_k,\tilde{u}_{k+1}\}$, we can utilize the Gram-Schmidt orthogonalization procedure to generate a set of mutually orthonormal vectors $\{\tilde{u}_{k+2},\dots,\tilde{u}_n\}$ from $\{t_{k+2},\dots,t_n\}$ by the relations

$$\tilde{u}_{k+2} = \frac{t_{k+2}}{|t_{k+2}|}, \tag{3.10a}$$

$$\tilde{t}_{k+i} = \sum_{j=k+2}^{k+i-1} (t^T_{k+i}\,\tilde{u}_j)\,\tilde{u}_j, \qquad \tilde{u}_{k+i} = \frac{t_{k+i} - \tilde{t}_{k+i}}{|t_{k+i} - \tilde{t}_{k+i}|}, \qquad i=3,\dots,n-k \tag{3.10b}$$

Now suppose that we desire to forget a known vector $\alpha^i \in \Lambda$, $\alpha^i \neq \alpha^m$, such that $\Lambda=\{\alpha^1,\dots,\alpha^{i-1},\alpha^{i+1},\dots,\alpha^m\}$. We let

$$y_i = \alpha^i - \alpha^m, \qquad Y_{m-1} = [y_1,\dots,y_{i-1},y_{i+1},\dots,y_{m-1}]. \tag{3.11}$$

Similarly as in the learning algorithm, we can check whether the set $\{y_1,\dots,y_{i-1}, y_{i+1},\dots,y_{m-1}\}$ spans a (k-1)-dimensional subspace or a k-dimensional subspace of L_{m-1} by computing rank(Y_{m-1}). If rank(Y_{m-1}) is equal to k, then y_i is a linear combination of $\{u_1,\dots,u_k\}$ (i.e., k=rank(Y_{m-1})). The left singular matrix $U=[u_1,\dots,u_n]$ obtained from the previous step still satisfies the requirements that $\{u_1,\dots,u_k\}$ is an orthonormal basis of L_{m-1} (i.e., L_{m-1}= Span(Y_{m-1})), and $\{u_{k+1},\dots,u_n\}$ is an orthonormal basis of $L^{\perp}_{m-1}$. If rank(Y_{m-1}) is equal to k-1, we let

$$t_k = y_i - \sum_{j=k+1}^{n} (y_i^T u_j)\, u_j. \qquad \tilde{u}_k = \frac{t_k}{|t_k|}. \tag{3.12}$$

Then $\{\tilde{u}_k,u_{k+1},\dots,u_n\}$ is an orthonormal basis of $L^{\perp}_{m-1}$. To find $U^+_{m-1}=[\tilde{u}_1,\dots,\tilde{u}_{k-1}]$ such that $\{\tilde{u}_1,\dots,\tilde{u}_{k-1}\}$ is an orthonormal basis of L_{m-1}, we utilize the tentative vectors $t_1,\dots,t_{k-1} \in R^n$ as follows:

$$\begin{aligned} t_1 &= [\,1,\ 0,\ \dots,\ 0,\ t_{1,k},\ \ t_{1,k+1},\ \ \dots,\ t_{1,n-1},\ \ t_{1,n}\,]^T, \\ t_2 &= [\,0,\ 1,\ \dots,\ 0,\ t_{2,k},\ \ t_{2,k+1},\ \ \dots,\ t_{2,n-1},\ \ t_{2,n}\,]^T, \\ &\dots \\ t_{k-1} &= [\,0,\ 0,\ \dots,\ 1,\ t_{k-1,k},\ t_{k-1,k+1},\ \dots,\ t_{k-1,n-1},\ t_{k-1,n}\,]^T. \end{aligned}$$

By construction, the vectors $t_1,\dots,t_{k-1}$ are linearly independent. We now need to ensure that the vectors $t_1,\dots,t_{k-1}$ are orthogonal to the vectors $\tilde{u}_k$ and $u_{k+1},\dots,u_n$. Again, we can arrive at a matrix equation to find the vectors $t_1,\dots,t_{k-1}$ by solving the equation

$$\begin{bmatrix} \tilde{u}_{k,k} & \tilde{u}_{k,k+1} & \cdots & \tilde{u}_{k,n-1} & \tilde{u}_{k,n} \\ u_{k+1,k} & u_{k+1,k+1} & \cdots & u_{k+1,n-1} & u_{k+1,n} \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ u_{n-1,k} & u_{n-1,k+1} & \cdots & u_{n-1,n-1} & u_{n-1,n} \\ u_{n,k} & u_{n,k+1} & \cdots & u_{n,n-1} & u_{n,n} \end{bmatrix} \begin{bmatrix} t_{i,k} \\ t_{i,k+1} \\ \cdot \\ \cdot \\ \cdot \\ t_{i,n-1} \\ t_{i,n} \end{bmatrix} = \begin{bmatrix} -\tilde{u}_{k,i} \\ -u_{k+1,i} \\ \cdot \\ \cdot \\ \cdot \\ -u_{n-1,i} \\ -u_{n,i} \end{bmatrix}, \quad i=1,\dots,k-1. \tag{3.13}$$

Since the vectors $t_1,\dots,t_{k-1}$ are linearly independent and orthogonal to the basis $\{\tilde{u}_k,u_{k+1},\dots,u_n\}$, we again apply the Gram-Schmidt orthogonalization procedure to generate a set of mutually orthonormal vectors $\{\tilde{u}_1,\dots,\tilde{u}_{k-1}\}$ from $\{t_1,\dots,t_{k-1}\}$ by the relations

$$\tilde{u}_1 = \frac{t_1}{|t_1|}, \tag{3.14a}$$

$$\tilde{\tau}_i = \sum_{j=1}^{i-1} (t_i^T \tilde{u}_j)\, \tilde{u}_j, \qquad \tilde{u}_i = \frac{t_i - \tilde{\tau}_i}{|t_i - \tilde{\tau}_i|}, \qquad i=2,\dots,k-1 \tag{3.14b}$$

Remark 3.3. The learning and forgetting procedure developed in the present subsection requires the inversion of a (k+1)×(k+1) matrix and an (n-k+1)×(n-k+1) matrix to solve the system equations (3.9) and (3.13). In applications, m may be large (k<m). This requirement may limit the effectiveness of the present learning procedure. In [7], we develop an iterative learning/forgetting algorithm for the eigenstructure method which avoids the complexities of high dimensional matrix inversions. This algorithm is also applicable to the present class of neural networks and can readily be implemented into the present procedure. We have omitted a discussion of this algorithm in the present paper because of space limitations. ■

IV. EXAMPLES

Example 4.1. The dimension of the system to be considered is n=10. Given are m=5 vectors specified by

$$\alpha^1 = \begin{bmatrix} -1 \\ 1 \\ -1 \\ 1 \\ 1 \\ 1 \\ -1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \quad \alpha^2 = \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \\ 1 \\ -1 \\ 1 \\ -1 \\ 1 \\ 1 \end{bmatrix}, \quad \alpha^3 = \begin{bmatrix} -1 \\ 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ 1 \\ -1 \\ 1 \\ -1 \end{bmatrix}, \quad \alpha^4 = \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \\ -1 \\ 1 \\ -1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \quad \alpha^5 = \begin{bmatrix} 1 \\ -1 \\ -1 \\ -1 \\ 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ -1 \end{bmatrix}.$$

It is desired that these vectors, $\Lambda=\{\alpha^1,\dots,\alpha^5\}$, be stored as stable output vectors of system (DN_τ).

1) Compute the $n\times(m-1)=10\times4$ matrix $Y_5=[y_1, y_2, y_3, y_4]$ where $y_i = \alpha^i - \alpha^5$, $1\le i\le4$.

2) Perform a singular value decomposition of Y_5 to obtain the matrices U, V and Σ_5 such that $Y_5 = U\Sigma_5V^T$. We have $k=\mathrm{rank}(\Sigma_5)=4$.

3) Compute the matrices $T_5^+ = u_1u_1^T + \dots + u_4u_4^T$, and $T_5^- = u_5u_5^T + \dots + u_{10}u_{10}^T$.

4) Choose the parameters $\tau_1 = 1.5$, $\tau_2 = \tau_{2,op} = 0.4951$ and compute

$$T_{op} = \tau_1 T_5^+ - \tau_{2,op} T_5^-, \qquad I_{op} = \tau_1 \alpha^5 - T_{op}\alpha^5.$$

Our simulation results verified that $\alpha^1,\dots,\alpha^5$ are stored as stable output vectors for the synthesized system (DN_{op}). In accordance with Remark 3.2, part 1, system (DN_{op}) has 15 additional stable output vectors given by

$$\beta^1=\begin{bmatrix}-1\\-1\\-1\\-1\\1\\1\\1\\-1\\-1\\-1\end{bmatrix},\ \beta^2=\begin{bmatrix}-1\\-1\\-1\\1\\-1\\1\\-1\\1\\-1\\-1\end{bmatrix},\ \beta^3=\begin{bmatrix}-1\\-1\\-1\\1\\1\\1\\-1\\1\\-1\\-1\end{bmatrix},\ \beta^4=\begin{bmatrix}-1\\-1\\1\\1\\-1\\1\\1\\-1\\-1\\-1\end{bmatrix},\ \beta^5=\begin{bmatrix}-1\\1\\-1\\-1\\1\\-1\\1\\-1\\1\\1\end{bmatrix},\ \beta^6=\begin{bmatrix}-1\\1\\-1\\1\\-1\\1\\-1\\1\\1\\1\end{bmatrix},\ \beta^7=\begin{bmatrix}-1\\1\\1\\1\\-1\\1\\-1\\1\\1\\-1\end{bmatrix},\ \beta^8=\begin{bmatrix}-1\\1\\1\\1\\1\\-1\\1\\-1\\1\\-1\end{bmatrix},$$

$$\beta^9=\begin{bmatrix}1\\-1\\-1\\-1\\-1\\1\\1\\-1\\-1\\-1\end{bmatrix},\ \beta^{10}=\begin{bmatrix}1\\-1\\-1\\-1\\1\\1\\-1\\1\\-1\\1\end{bmatrix},\ \beta^{11}=\begin{bmatrix}1\\-1\\-1\\1\\-1\\1\\-1\\1\\-1\\-1\end{bmatrix},\ \beta^{12}=\begin{bmatrix}1\\-1\\-1\\1\\1\\1\\-1\\1\\-1\\-1\end{bmatrix},\ \beta^{13}=\begin{bmatrix}1\\1\\-1\\-1\\-1\\-1\\1\\-1\\1\\1\end{bmatrix},\ \beta^{14}=\begin{bmatrix}1\\1\\-1\\1\\1\\1\\-1\\1\\1\\1\end{bmatrix},\ \beta^{15}=\begin{bmatrix}1\\1\\1\\1\\-1\\-1\\1\\-1\\1\\-1\end{bmatrix}.$$

Also, by Remark 3.2, part 2, we determine that $L_a \cap B^n=\{\alpha^1,\dots,\alpha^5\}$. ■

Remark 4.1. In Table 4.1, the total number of stable output vectors of the synthesized system for different values of τ are listed and compared with the other three methods discussed in [1]-[4]. In this table, "total number of stable output vectors" signifies the total number of stable output vectors of system (DN) with T and I synthesized by means of the enumerated methods while "total number of unstable output vectors" denotes the total number of vectors in the set $\{\alpha^1,\dots,\alpha^5\}$ which were actually unstable. For each α^i, $1\le i\le 5$, there are ten adjacent vectors which differ by one component from α^i. For each α^i, we tested the adjacent vectors to see whether they are contained in the domain of attraction of α^i. In Table 4.1, "total number of adjacent vectors in the domains of attraction of α^i, $1\le i\le 5$" denotes the total number (out of $10\times 5=50$) of these adjacent vectors which belong to the corresponding stable output vector α^i, $1\le i\le 5$. This test was applied to systems obtained by the present method, for different values of the parameter τ, $\tau\le\tau_{2,op}$, and also to systems obtained by the three other methods (i.e., Hopfield's method [1], Personnaz's method [2], and the continuous version of the present class of neural network designed by the eigenstructure method [4]) . ■

Remark 4.2. The results listed in Table 4.1 suggest the following.

1) Using the present method, the vectors $\alpha^1,\dots,\alpha^5$ are always stored as stable output vectors. Also, as τ_2 increases, the number of spurious stable output vectors decreases. Furthermore, when $\tau_2\le\tau_{2,op}$, the extent of the domain of attraction of each α^i, $1\le i\le 5$, increases as τ_2 increases.

2) When compared to the previous results developed in [1]-[2], the present results appear to offer several advantages. Specifically, the present synthesis method guarantees to store all of the desired vectors as asymptotically stable equilibrium points whereas the results in [1]-[2] do not give such assurances. Also, the present design method offers a controlling mechanism to reduce the number of spurious states and to estimate the extent of basins of attractions of the stored vectors whereas the results in [1]-[2] do not guarantee this.

3) The present results are based on similar theoretical principles as the results developed in [4] (for continuous-time neural networks described by a system of discontinuous first-order ordinary differential equations). Therefore, as expected, the attributes and liabilities of the present results and those given in [4] are similar. However, the results developed herein offer

several advantages over those developed in [4]. Specifically, the present results have learning and forgetting capabilities whereas this is not the case for the results given in [4]. Also, the present results can be implemented by specialized digital hardware. Finally, when implemented by a serial processor, the present results offer superior means of simulating corresponding continuous-time neural networks. ■

Example 4.2. Continuing Example 4.1, we utilize the learning algorithm presented in Section III.B, to store into the network the vector $\alpha^6 = [1\ 1\ 1\ 1\ -1\ -1\ 1\ 1\ -1\ -1]^T$ while retaining the original vectors α^i, $1 \le i \le 5$. In addition to the stable output vectors α^i, $1 \le i \le 6$ given in Example 4.1, the resulting network stores nineteen additional spurious stable output vectors.

Simulations verified that the desired indicated memories are indeed stored into the synthesized network. Applying the forgetting algorithm to the above network, we successfully delete the memories α^3 and α^6 as verified by simulations. In addition to α^i, i=1, 2, 4, and 5, the resulting network also stores eleven spurious stable output vectors. ■

V. CONCLUSIONS

In the present paper we established a *qualitative theory* and we developed an efficient *synthesis procedure* with iterative learning and forgetting capabilities for a class of discrete-time neural networks described by a system of discontinuous first order ordinary difference equations. When compared with previous results [1]-[4], the method advanced herein offers several advantages (see Remark 4.2). To demonstrate the applicability of the present results, we considered two specific examples.

REFERENCES

[1] J. J. Hopfield, “Neural Networks and Physical Systems with Emergent Collective Computational Abilities”, *Proc. Natl. Acad. Sci.*, Vol. 79, April 1982, pp. 2554-2558.

[2] L. Personnaz, I. Guyon and G. Dreyfus, “Information Storage and Retrieval in Spin-Glass like Neural Networks”, *J. Physique Lett.*, Vol. 46, April 1985, pp. L359-L365.

[3] J. H. Li, A. N. Michel and W. Porod, “Qualitative Analysis and Synthesis of a Class of Neural Networks”, *IEEE Transactions on Circuits and Systems*, Vol. 35, No. 8, August 1988, pp. 976-986.

[4] J. H. Li, A. N. Michel and W. Porod, “Qualitative Analysis and Synthesis of a Class of Neural Networks: Variable Structure Systems with Infinite Gains”, *IEEE Transactions on Circuits and Systems*, Vol. 36, No. 5, May 1989, pp. 713-731.

[5] W. Hahn, *Stability of Motion*, Springer-Verlag, Berlin, 1967.

[6] V. I. Utkin, “Variable Structure Systems with Sliding Modes”, *IEEE Transactions on Automatic Control*, Vol. 22, No. 2, April 1977, pp. 212-222.

[7] G. Yen and A. N. Michel, “A Learning and Forgetting Algorithm in Associative Memories: The Eigenstructure Method”, to appear in the *Proceedings of the 30th IEEE Conference on Decision and Control*, Brighton, United Kingdom, December 1991.

Table 4.1. Comparison of the Qualitative Properties for four Different Synthesis Techniques

	Hopfield Method [1]	Personnaz Method [2]	Continuous Version $\tau=0$ [4]	Continuous Version $\tau=\tau_{op}$ [4]	Discrete Version $\tau_1=1.5$, $\tau_2=0$	Discrete Version $\tau_1=1.5$, $\tau_2=\tau_{2,op}$
total number of stable output vectors	4	122	58	26	48	20
total number of unstable output vectors	5	0	0	0	0	0
total number of adjacent vectors in the domains of attraction of $\alpha^i, 1 \le i \le 5$	0	30	40	40	38	42

Hidden Layer Configurations of Neural Networks and Their Learning Performance

Jia-Yuan Han[*], Xun Lei[*], John Wu[+], and Gui-Qiang Chen[++]

Abstract

The learning performance of three kinds of hidden layer configurations of back-propagation neural networks, that is, geometric series configuration (GSC), arithmetic series configuration (ASC) and equal-width configuration (named as BPC), is addressed in this manuscript. After providing the description and formula of multi-layer neural nets and their nonlinear active functions of three configurations, their learning accuracies and training speeds are compared, respectively. Computer simulations show that the back-propagation neural network with the GSC has the fastest learning speed and the highest accuracy. A new approach to improve the learning speed of a neural network of back-propagation is proposed.

1. Introduction

Artificial neural networks have been successfully applied to various fields, such as, signal processing, pattern recognition, control, machine intelligence, and many other areas [1,2]. One of the main features of neural networks is their powerful learning capability [1-3]. Back-propagation neural network is one of the widely applied neural networks with strong learning capability [4-6]. It has been successfully applied to many areas.

This work was supported in part by the Education and Research Initiation Fund at Illinois Institute of Technology.

[*] Electrical and Computer Engineering Department, Illinois Institute of Technology, Chicago, IL 60616

[+] Electrical Engineering Department, Auburn University, Urburn, AL 36849-5201

[++] The Department of Mathematics, The University of Chicago, Chicago, IL 60637

However, one of the main drawbacks of back-propagation networks is their slow learning speed during the training process, which affects their performance and application range [4-6]. Another main drawback of back-propagation is its arbitrary choices regarding the number of hidden units and layers. This makes its internal structure very flexible. Many researchers have tried to improve the training speed of the neural networks [7-11]. However, the performance of a back-propagation neural network with a different internal configuration has not been investigated before. This motivates our investigation on hidden layer structures and their effects on learning speed and accuracy.

2. Main Results

2.1 Back-propagation Neural Network with Multiple Layers

The dynamics of an artificial neural network with 2 layers has been exposed profoundly in [12]. Consider an artificial neural network with M layers consisted of M-1 hidden layers and output layers (notice that the input layer is not counted as one layer), The dynamics in the search phase of the neural network is further expressed as follows. The notations used below follow the notations in [12]. For k = 1, we have

$$\mathrm{Net}_i^{k-1} = \sum_{j=1}^{a_1} W_{ij}^{k-1} X_j + \Theta_i^{k-1}, \tag{1}$$

$$\mathrm{Out}_i^{k-1} = F(\mathrm{Net}_i^{k-1}), \tag{2}$$

where $i = 1, \ldots, a_0$. In the above formulas, $\{W_{ij}^{k-1}\}$ denotes the value of a weight from neuron i in hidden layer k-1 to neuron j in hidden layer k, $\{\mathrm{Out}_j^{k-1}\}$ denotes the value of neuron j in the hidden layer k-1, F denotes the nonlinear activation function, a_0 denotes the number of neuron units corresponding to input patterns, $\{\Theta_j^{k-1}\}$ denotes the internal offsets of the neurons in the hidden layer k-1, and $\{X_i\}$ denotes the input patterns. $\{Y_{ij}^M\}$ denotes the activation value in the output layer M.

For k = 2, 3,..., M-1, we have

$$Net_i^k = \sum_{j=1}^{a_{k+1}} W_{ij}^k \, out_j^{k-1} + \Theta_i^k, \tag{3}$$

$$Out_i^k = F(Net_i^k), \tag{4}$$

where i = 1,..., a_k. For the output layer, that is, k = M, we have

$$Net_i^M = \sum_{j=1}^{a_{out}} W_{ij}^k \, Out_j^{k-1} + \Theta_i^k, \tag{5}$$

$$Y_i^M = Out_i^k = F(Net_i^M), \tag{6}$$

where i = 1,..., a_k, and a_{out} is the number of neurons in the output layer.

The error-rule of the N distinct training patterns, consisting of N pairs of outputs and target patterns $\{Y_i(n), t_i(n)\}$ is expressed as follows.

$$E = \sum_{n=1}^{N} \sum_{i=1}^{a_{out}} [t_i(n) - Y_i^M(n)]^2$$

$$= \sum_{n=1}^{N} \sum_{i=1}^{a_{out}} [t_i(n) - F(Net_i^M(n))]^2, \tag{7}$$

When the N-th pattern is available for training, the following equation illustrates weight updating in a supervised learning.

$$W_{ij}^k(n) = W_{ij}^k(n-1) + \Delta W_{ij}^k(n), \tag{8}$$

$$\Delta W_{ij}^k(n) = \mu \, (t_i(n) - Y_i^M(n)) \, Out_i^k(n), \tag{9}$$

The weight updating of the output layer is as follows.

$$\Delta W_{ij}^k(n) = \mu \, (t_i(n) - Y_i^M(n)) Out_i^{k-1}(n)$$

$$= \zeta F'(Net_i^k(n))(t_i(n) - Y_i^M(n)) Out_i^{k-1}(n) \qquad k = M \tag{10}$$

The formula of weight-updating for hidden layers can be described as follows.

$$\Delta W_{ij}^{k-1}(n) = \zeta F'(Net_i^{k-1}(n)) \, [\sum_j F'(Net_i^k(n) \, (t_j(n) - Out_i^k(n)) \, W_{ji}^k(n-1)] \, X_j(n),$$

where k = M-1, M-2,..., 1. (11)

2.2 Three Kinds of Configurations of Hidden Layers

So far most of the back-propagation networks are constructed with the same number of neurons at each layer except the input layer and the output layer, as shown in Fig. 1. This kind of configuration is refered to as back-propagation configuration (BPC) for convenience. Since the input and output layers, depending on the number of inputs and outputs, are fixed for all back-propagation networks , the differences among various configurations are the internal layers of the back-propagation networks. Here two new configurations of back-propagation networks are proposed. The first one has the hidden layers of neurons with geometric series distribution, called the hidden layers with GSC, as shown in Fig. 2. The second one has the hidden layers of neurons with arithmetic series distribution, called the hidden layers with ASC, as shown in Fig. 3.

In order to compare their performances, the back-propagation networks with three hidden layers are constructed for each of GSC, ASC, and BPC. The comparison has to be performed based on the same number of layers and the same number of total neurons in the hidden layers in each of the configurations. We chose that the number of layers is 3 and the total number of neurons is 21. The detailed structures are shown below.

1. Back-propagation neural network with GSC:

$$a_1 = 3,\ a_2 = 6, \text{ and } a_3 = 12$$

2. Back-propagation neural network with ASC:

$$a_1 = 5,\ a_2 = 7, \text{ and } a_3 = 9$$

3. Back-propagation neural network with BPC:

$$a_1 = 7,\ a_2 = 7, \text{ and } a_3 = 7$$

2.3 Computation Time and Training Time

After estimating the number of operations in each training cycle for the above three configurations respectively, a comparison of computation cost of three configurations is performed and shown in Fig. 4. The results show that the neural network with GSC takes the least number of operations in each training cycle and the neural network with BPC requires the longest computation time in each training cycle.

The training time not only depends on the computation cost in each training cycle, but also depends on the total number of training cycles. By computer simulations, we have shown that with the same number of training cycles the neural network with GSC converges to the trained pattern much faster than any other configurations. The neural network with ASC has the second convergence speed. The results with one training cycle are shown in Table 1.

2.4 Accuracy of Trained Neural Network

In order to compare the accuracy of a trained neural network with three kinds of configurations, computer simulations were performed. The computer simulations show the following:

(1) If a neural network of the same type of configuration has a larger number of neurons, it has higher accuracy than the neural network with less neurons after the same number of training steps. Simulations of back-propagation neural networks with ASC are shown in Table 2. They both have 3 layers. One has 12 neurons and the other has 15 neurons. After one training cycle, the errors for ramp inputs show that the neural network with more neurons yields a smaller error than the one with less neurons.

(2) The neural network with GSC has the highest accuracy and the neural network with ASC has the second highest accuracy after the same number of training cycles. Computer simulations are also shown in Table 1. All three neural networks have 3 layers and 21 neurons. The errors after one training cycle are shown in the table.

3. Conclusion

From the above research, one can see that the training speed and accuracy of a back-propagation neural network depend on its internal structures. Computer simulations show that the neural networks with GSC have better performance than other ones with ASC or BPC. The neural networks with GSC are recommended as better configurations for back-propagation neural networks.

Acknowledgments

The authors are grateful to Prof. T. Wong for his continuous support.

References

[1] R. P. Lippmann, "An Introduction to Computing with Neural Nets," IEEE ASSP Magazine, Vol. 4, No. 2, pp. 4-22, 1987.

[2] D. E. Rumelhart, G.E. Hinton, and R. J. Williams, "Learning Internal Representation by error propagation," *Parallel Distribution Processing,* Vol. 1, D.E. Rumelhart and J. L. Mcclelland (eds.) Cambridge, MA: MIT Press, pp. 318-362, 1986.

[3] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuron-like Adaptive Elements that Can Solve Difficult Learning Control Problem," IEEE Transactions on Systems, Man and Cybernetics Vol. SMC-13, pp. 834-846.

[4] P. J. Werbos, "Back Propagation: Past and Future," Proc. 1988 IEEE Conf. Neural Networks, July 1988, Vol. 1, pp. 343-353

[5] R. Hecht-Nielsen, *Neurocomputing,* Addison-Wesley, Reading, MA, 1990

[6] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Representation by Backpropagation Error," Nature, Vol. 323, pp. 533-536, 1986.

[7] T. Kohonen and E. Oja, "Fast Adaptive Formation of Orthogonalizing Filters and Associative Memory in Recurrent Network of Neuron-Like Elements, " Biol. Cybernetics, Vol. 21,pp. 85-95, 1976.

[8] J. Moody and C. Darken, "Fast Learning in Networks of Locally-tuned Processing Units, " Neural Computation, Vol. 1, No.2, pp.281-294, 1989.

[9] S. Y. Kung and J. N. Hwang " An Algebraic Projection Analysis for Optimal Hidden Units Size and Learning Rates in Back-Propagation Learning," IEEE First International Conference on Neural Network, Vol.1, pp. 363, 1987.

[10] R.P.Brent, "Fast Training Algorithms for Multi-Layer Neural Nets," Technical Report NA-90-03, Department of Computer Science, Stanford University, Mar. 1990, Extended Abstract in Proc. First Australian Conf. Neural Networks (University of Sydney, Australia). Jan. 1990, pp.97-98.

[11] R. P. Brent," Fast Training Algorithms for Multi-layer Neural Nets," IEEE Transactions on Neural Networks , Vol.2, No.3, pp.346-354, May 1991.

[12] R. Hecht-Nielsen, "Counterpropation Networks " Applied Optics, Vol. 26, No.23, Dec. 1987.

Table 1. The Comparison of Learing Capability of the Neural Network with Three Kinds of Configurations of Hidden Layer

Configuration of Hidden Layer /Index	Training Times	Numbers of Neuron Units	Numbers of Layer	Error (Parabola Input)
BPC	1	21	3	1.6981315E-17
ASC	1	21	3	1.4971536E-24
GSC	1	21	3	9.9235339E-25

Table 2. The Neural Network with Different Neuron Units

Configuration of Hidden Layer /Index	Training Times	Numbers of Neuron Units	Numbers of Layer	Error (Ramp Input)
The Hidden Layer with ASC	1	15	3	9.8386475E-18
	1	12	3	5.513578E-16

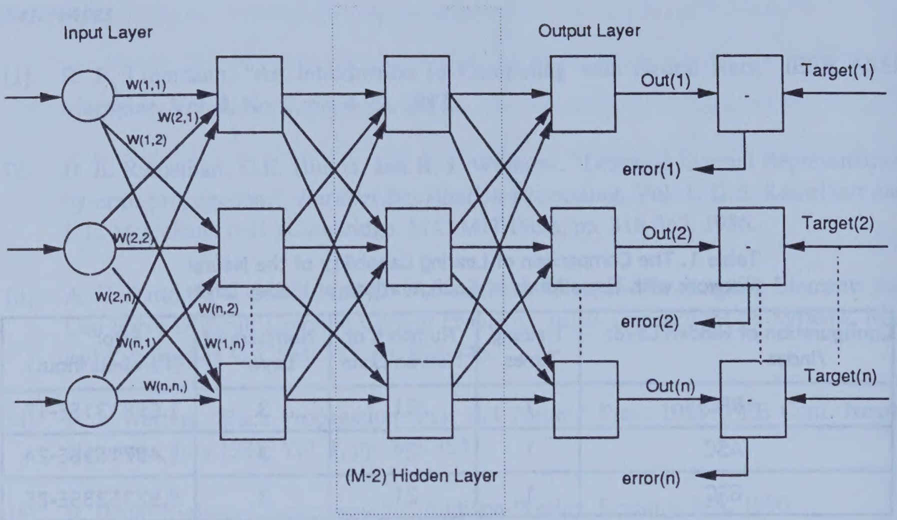

Input Layer
W(1,1)
W(2,1)
W(1,2)
W(2,2)
W(2,n)
W(n,2)
W(n,1)
W(1,n)
W(n,n,)
(M-2) Hidden Layer
Output Layer
Out(1)
Out(2)
Out(n)
-
-
-
Target(1)
Target(2)
Target(n)
error(1)
error(2)
error(n)

Fig. 1 M-layer Neural Network of the Hidden Layer With BPC

(M-Layer Backpropagation Neural Network)

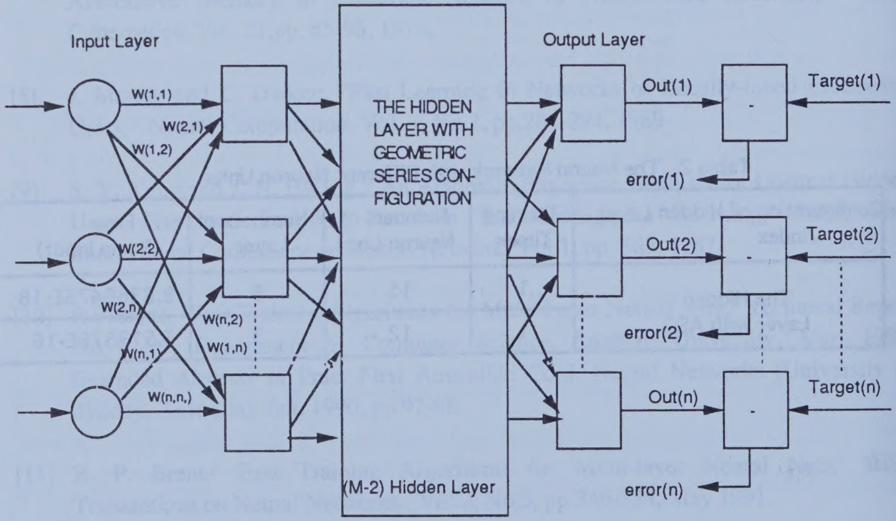

Input Layer
W(1,1)
W(2,1)
W(1,2)
W(2,2)
W(2,n)
W(n,2)
W(n,1)
W(1,n)
W(n,n,)
THE HIDDEN LAYER WITH GEOMETRIC SERIES CON-FIGURATION
(M-2) Hidden Layer
Output Layer
Out(1)
Out(2)
Out(n)
-
-
-
Target(1)
Target(2)
Target(n)
error(1)
error(2)
error(n)

Fig. 2 M-layer Neural Network of the Hidden Layer With GSC

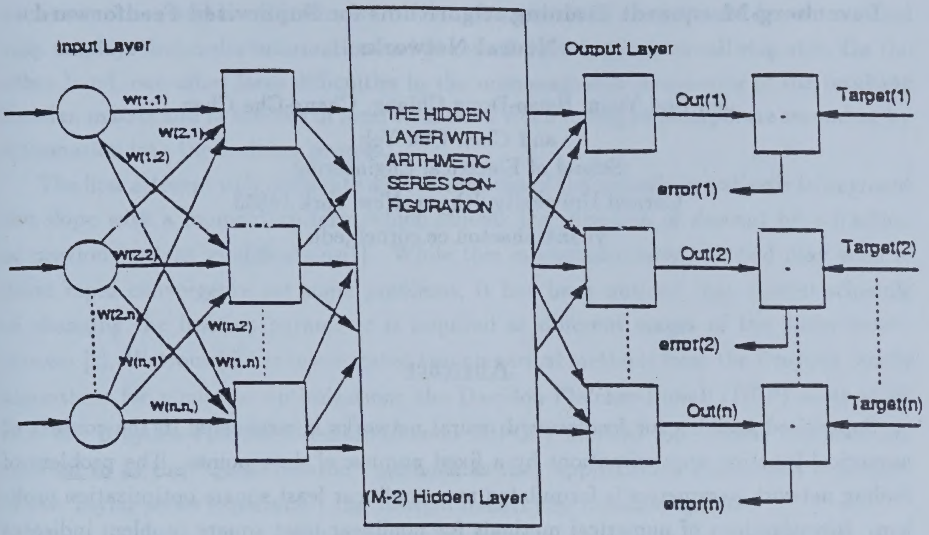

Input Layer
W(1,1)
W(2,1)
W(1,2)
W(2,2)
W(2,n)
W(n,2)
W(n,1)
W(1,n)
W(n,n)
THE HIDDEN LAYER WITH ARITHMETIC SERIES CONFIGURATION
(M-2) Hidden Layer
Output Layer
Out(1)
Out(2)
Out(n)
Target(1)
Target(2)
Target(n)
error(1)
error(2)
error(n)

Fig.3 M-layer Neural Network of the Hidden Layer With ASC

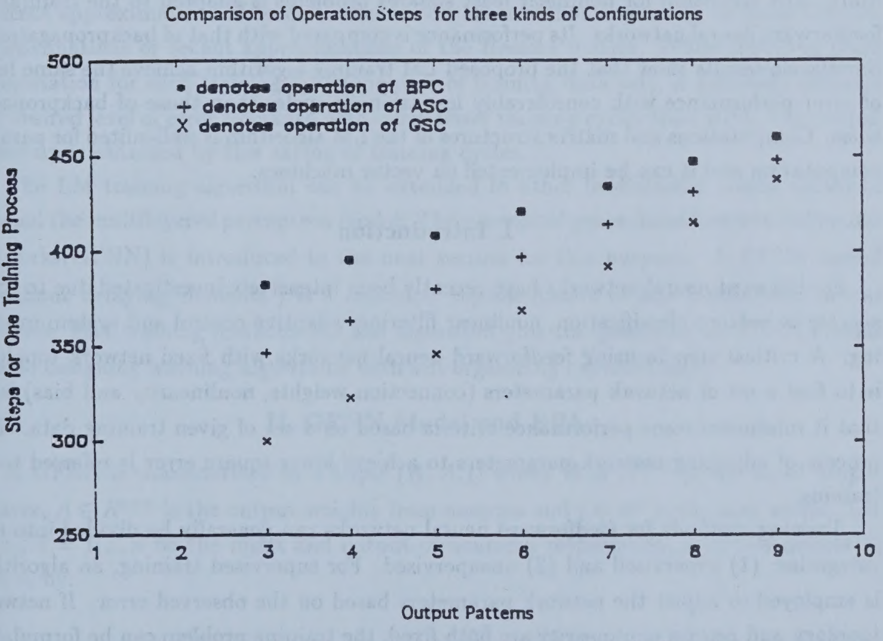


Fig.4 Comparison of Operation Steps for Three Kinds of Configurations

Levenberg-Marquardt Training Algorithms for Supervised Feedforward Neural Networks

Jen-Lun Yuan, Hsiao-Dong Chiang, Cheng-Che Chen
and Chen-Chu Yeh
School of Electrical Engineering
Cornell University, Ithaca, New York 14853
yuan@phaeton.ee.cornell.edu

Abstract

Supervised training for feedforward neural networks is considered in the context of numerical function approximations by a fixed number of data points. The problem of finding network parameters is formulated as a nonlinear least square optimization problem. Investigations of numerical methods for nonlinear least square problem indicates that algorithms which exploit both least square structures and approximations of the second order derivatives may serve efficiently as training procedures. The Levenberg-Marquardt algorithm for nonlinear least squares problems is adapted to the training of feedforward neural networks. Its performance is compared with that of backpropagations. Simulation results show that the proposed LM training algorithm achieve the same level of error performance with considerably less training cycles than those of backpropagations. Computations and matrix structures of the LM algorithm is well-suited for parallel computation and it can be implemented on vector machines.

I. Introduction

Feedforward neural networks have recently been intensively investigated due to their success in pattern classification, nonlinear filtering, adaptive control and system modeling. A critical step in using feedforward neural networks with fixed network topology is to find a set of network parameters (connection weights, nonlinearity and bias) such that it minimizes some performance criteria based on a set of given training data. The process of adjusting network parameters to achieve lower square error is referred to as training.

Training methods for feedforward neural networks can generally be divided into two categories: (1) supervised and (2) unsupervised. For supervised training, an algorithm is employed to adjust the network parameters based on the observed error. If network topology and neuron nonlinearity are both fixed, the training problem can be formulated as a nonlinear optimization problem with the square error as the objective function. The widely-used backpropagation algorithm(BPA) minimizes square errors by the steepest descent search combined with local structures of feedforward networks. By backward substitution, BPA efficiently computes gradients of objective functions with respect to each connection and bias[1]. However, it is well-known that the steepest descent method only employs first-order information and is not efficient due to its small step size. On the other hand, one often faces difficulties in the unmanageable complexity of the resulting Hessian matrix and in the loss of local structures when trying to incorporate second-order information into the training process.

The first attempt to incorporate approximations of the second derivative is to augment the slope with a momentum term which adjusts the direction of descent by a fraction of previous weight modifications[1]. While this momentum term method may achieve more rapid convergence for some problems, it has been noticed that careful schedule of changing the fraction parameter is required at different stages of the optimization process [2]. Watrous [3] has investigated two numerical methods from the Broyden family algorithms for nonlinear optimization: the Davidon-Fletcher-Powell (DFP) method [4] and the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method [5]. These methods are referred to as the "Quasi-Newton" methods as they approximate the second derivative of the Taylor series expansion (the Hessian matrix) by iterative updates.

This paper describes a training algorithm based on a nonlinear least square optimization method proposed by Levenberg and Marquardt(LM)[7]. The basic idea behind this algorithm is to explore the structure of square error objective functions that give efficient approximations of second-order information without resorting to quasi-Newton approximations or secant approximations of the Hessain matrix. While requiring more computation for each training cycle(one pass of training data set), it generally achieves the desired level of error tolerance with much fewer training cycles than BPA. The overall speed-up is obtained by this saving of training cycles.

The LM training algorithm can be extended to other feedforward neural networks beyond the multilayered perceptron model. The concept of generalized feedforward neural networks(GFNN) is introduced in the next section for this purpose. A GFNN model (without delaying elements) is a canonical representation of any feedforward neural network. Fast training methods like LM algorithm plus the generality of GFNN models enable designing learning algorithms with self-organizing capabilities[9].

II. GFNN Model and BPA

A GFNN is characterized by a triple $(W, A, \underline{t})$ where $W \in R^{m \times n}$ is the input weight matrix, $A \in R^{n \times n}$ is the output weights from neurons and $\underline{t} \in R^n$ is the bias vector. Let u_i, z_i, $i = 1, .., n$ be the input and output of neuron i respectively, a GFNN defines u_i and z_i by:

$$u_i = \sum_{j=1}^{n} w_{ij} x_j + \sum_{j=1}^{i-1} a_{ij} z_j - t_i \quad (1)$$

$$z_i = \sigma(u_i) \quad (2)$$

where σ is the nonlinear activation function such as $1/(1 + e^{-x})$.

The class of training data is a finite set $T = \{(x^k, y^k) : k = 1, .., N\}$ where $x^k = [x_1^k, .., x_m^k]'$ is an m-vector representing the input to the neural network, and $y^k = [y_1^k, .., y_p^k]'$ is the output associated with x^k. y^k is usually called the desired output or the reference output.

A GFNN operates as follows: at any time instance k, where $k = 1, .., N$ the x-vector of training data is fed into the input variables x. Each neuron in GFNN then takes its inputs from input variables x_j (if available) and the outputs z_j of neurons $j = 1, .., i-1$ to form its net input by a linear weighted sum of (1). The output of neuron j is given by the nonlinear function $\sigma(\cdot)$ of (2).

Theoretically, it can be shown that a GFNN can be used to approximate ,to any degree of accuracy, any function in the space $C(R^m, R^p)$ provided that a sufficiently large number of nodes and links are given in a GFNN[6].

Theorem[GFNN] Let I_m denote the closed m_dimensional hypercube $[0,1]^m$. Let $C(I_m)$ denotes the space of continuous functions on I_m. Assume that σ is a sigmoidal function defined above, then the set of functions defined by a GFNN (with single output) of the form:

$$G(x) = \sigma\left(\sum_{j=1}^{m} w_{ij} x_j + \sum_{j=1}^{i-1} a_{ij} z_j - t_n\right) \tag{3}$$

is dense in $C(I_m)$.

The previous theorem only shows that there exists a neural network provided that the training data T are taken from a set of data generated by some function in the space $C(I_m)$. In practice, it is difficult to find such a network, it is therefore still an open problem as to find algorithms which guarantee the performance promised by this theorem.

Heuristically, a neural network can be constructed first by specifying a network architecture and then by training the network to minimize the L_2-norm of the difference vector of the training data and the function of the form $G(\cdot)$ lying in the function space $C(I_m)$:

$$\min_{w,a,\underline{t}} J(w, a, \underline{t}) := \frac{1}{N} \sum_{k=1}^{N} J^k(x^k, w, a, \underline{t}) \tag{4}$$

where

$$J^k(x^k, w, a, \underline{t}) = \sum_{h=1}^{p} (y_h^k - z_{n-p+h}^k)^2 \tag{5}$$

The BPA updates the connection weights and bias by the gradient information:

$$\Delta w_{il} = -r_w \frac{\partial J}{\partial w_{il}} \tag{6}$$

$$
\begin{aligned}
\triangle a_{ij} &= -r_a \frac{\partial J}{\partial a_{ij}} && (7)\\
\triangle t_i &= -r_t \frac{\partial J}{\partial t_i} \qquad i,j = 1,..,n; \quad l = 1,..,m && (8)
\end{aligned}
$$

where r_a, r_w and r_t are the learning rates(step size parameters) for updating the input weights, output weights and bias, respectively.

The disadvantage of this update rule (by BPA), as mentioned before, is that only first-order information is employed, thus may cause a slow convergence rate. It motivates the use of second-order information in the searching process. A direct generalization of error propagation which utilizes the specific structure of square error is the Levenberg-Marquardt method, which can be considered as a compromise between these two extremes: gradient search and the Newton method.

III. LM Training Algorithm

The quadratic error function in (5) motivates the idea of the nonlinear least square solution. The nonlinear least-square problem arises mostly from data-fitting applications. The neural network training problem can be described as a data-fitting problem by (4)-(5).

Consider the following objective function adopted from (4):

$$J(x) = \sum_{i=1}^{N} r_i^2(x_1, x_2, .., x_M) \tag{9}$$

where $r_i(\underline{x}) = J^i(\underline{x}^i, w, a, \underline{t})$, M is the number of network parameters (including input weights, output weights and bias) and N is the number of training data in T multiplied by the output dimension (For simplicity, we consider $p = 1$; single output case). Let

$$\underline{x} = \begin{bmatrix} x_1, & \cdots, x_M \end{bmatrix}'$$

represent the weight vector. Define

$$R = \begin{bmatrix} \frac{\partial r_1}{\partial x_1} & \frac{\partial r_1}{\partial x_2} & \cdots & \frac{\partial r_1}{\partial x_M} \\ \frac{\partial r_2}{\partial x_1} & \frac{\partial r_2}{\partial x_2} & \cdots & \frac{\partial r_2}{\partial x_M} \\ \vdots & & & \vdots \\ \frac{\partial r_N}{\partial x_1} & \frac{\partial r_N}{\partial x_2} & \cdots & \frac{\partial r_N}{\partial x_M} \end{bmatrix} = \begin{bmatrix} s_1' \\ s_2' \\ \vdots \\ s_N' \end{bmatrix} \qquad \underline{r}(\underline{x}) = \begin{bmatrix} r_1(\underline{x}) \\ r_2(\underline{x}) \\ \vdots \\ r_N(\underline{x}) \end{bmatrix} \tag{10}$$

The Levenberg-Marquardt method for nonlinear least square problem suggests an iterative search by the equation[7]:

$$
\begin{aligned}
\underline{x}^{k+1} &= \underline{x}^k - (R_N^{k\,\prime} R_N^k + \mu^k I)^{-1} R_N^{k\,\prime} \underline{r}(\underline{x}^k) && (11)\\
k &= 1, 2, .., \text{ number of training cycles.}
\end{aligned}
$$

where we define:

$$\begin{aligned} R_i^k &= \left[\, s_1^{k\prime}, \ \cdots \ s_i^{k\prime}, \ 0, \ \cdots \ 0 \,\right]' \\ i &= 1, 2, .., N \end{aligned}$$

where s_i is the partial error obtained by error-propagation (6)-(8) after the i_th training data has been passed through the neural network.

Let

$$T_i^k = R_i^{k\prime} R_i^k + \mu^k I,$$

then

$$T_{i+1}^k = T_i^k + s_{i+1}^k{}' s_{i+1}^k \tag{12}$$

where μ^k is the learning rate at cycle k.

By the Sherman-Morrison formula:

$$\begin{aligned} T_{i+1}^k{}^{-1} &= [\, T_i + s_{i+1}^k{}' s_{i+1}^k \,]^{-1} \\ &= (T_i^k)^{-1} - \frac{(T_i^k)^{-1} s_{i+1}^k s_{i+1}^k{}' (T_i^k)^{-1}}{1 + s_{i+1}'(T_i^k)^{-1} s_{i+1}} \end{aligned} \tag{13}$$

with

$$T_0^k = \mu^k I$$

Equation (13) can be computed recursively in "batch" mode(i.e., the weight vector x is updated only when the whole training set has been sent to the neural network, and the algorithm updates x once for each pass of T). Combining (10)-(13) with error-propagation for GFNN in (6)-(8), we have the following algorithm:

Algorithm[LM Training]

Input: training set T, a GFNN $(w, a, \underline{t})$. Set $k = 0$, $(T_0^k)^{-1} = I/\mu^k$, $i = 0$.

(Repeat Step 1 - Step 4 for $k = 1, .., $ number of training cycles.)

Step 1:(Feedforward pass)

Compute outputs of GFNN by:

$$\begin{aligned} u_i &= \sum_{j=1}^{n} w_{ij} x_j + \sum_{j=1}^{i-1} a_{ij} z_j - t_i \\ z_i &= \sigma(u_i) \end{aligned}$$

Step 2(Error Propagation)

Compute first-order information by (6)-(8):

$$s_{i+1} = \left[\ \frac{\partial J_i}{\partial w} \ \ \frac{\partial J_i}{\partial a} \ \ \frac{\partial J_i}{\partial t} \ \right]^T$$

where J_i is define in (5).

Step 3(Update LM iterative step)

Compute T_{k+1}^{-1} by:

$$(T_{i+1}^k)^{-1} = (T_i^k)^{-1} - \frac{(T_i^k)^{-1} s_{i+1}^k s_{i+1}'^k (T_i^k)^{-1}}{1 + s_{i+1}'(T_i^k)^{-1} s_{i+1}}$$

Update R_{i+1}^k and $\underline{r}(\underline{x}^k)$.

Step 4(Testing for batch condition)

If $i = N$ update network parameters by :

$$x_{k+1} = x_k - {T_N^k}^{-1} (R_N^k)' r(x^k)$$

and set $i = 0,\ \mu = 0$

else update μ^k and let

$$i = i + 1$$

$k = k + 1$, go to Step 1.

The choice of step size parameter μ used in LM training can be found in [7]. The Moré method[8] is used in our implementation of LM training.

IV. Simulation Results

The LM training algorithm applied to GFNN model has been implemented in C source codes running on SPARC stations. The program has been tested for two cases. In the first case, the historical XOR problem with 3 variables is tested on a three node two-layer network with sigmoidal nonlinearity. Its time performance is compared with that of BPA (with adaptive step size). The error tolerance level for both algorithms is set to 10^{-6}. Figure 1 shows that the curves of the progress of average square error for LM training with three different initial conditions compared with that of BPA training. The error of LM training converges at around 100 training cycles while BPA takes almost 3000 cycles to achieve the same level of error. In the second case, we applied both algorithms to a power system load modeling problem with 100 training data. The same architecture of a 3-layer model with (4,4,1) neurons is used for both algorithms. The error tolerance is set to 10^{-4}. Figure 2 shows that the error curve associated with the LM training algorithm ,with two different initial conditions, drop to the level of 1.5×10^{-4} within 100 training cycles. By contrast, the errors curve from BPA still stays above 4×10^{-4} by the end of 10000 training cycles.

V. Concluding Remarks

We have shown that by taking into account second order information in the training process, the convergence performance can be improved in the sense that fewer training cycles are required to achieve the same level of error performance as those achieved by backpropagations. In particular, we have proposed an algorithm based on Levenberg-Marquardt nonlinear least square method which gives the improved performance.

The speed-up gained by LM training is obtained at the expense of: (i) more computation performed in each training cycle as compared with those of BPA, and (ii) more memory used for holding the intermediate results generated by the matrix computations. These two problems may restrict the practical use of LM-training to small or medium sized problems. However, these difficulties can be alleviated by currently developing parallel computation techniques and by the large-scale optimization methods.

Further research along this direction includes implementation of the LM algorithm on vector machine, parallel computers and by techniques which allows computations of large matrices. This may lead to a training paradigm which is efficient and is capable of solving large scale problems for practical applications of supervised feedforward neural networks.

VI. Acknowledgement

Thanks to Dr. Jin Lu for valuable discussions on nonlinear least square problems and LM algorithms.

References

[1] Rumelhart, D. and McClelland, J.,*Parallel Distributed Processing*, The MIT Press, 1986.

[2] Fahlman, S.E., "Faster Learning Variations on Backpropagation: An Empirical Study", *Proc. of the 1988 Connectionist Models Summer School,* San Mateo, CA, 1988.

[3] Watrous, R.L., "Learning Algorithms for Connectionist Networks: Applied Gradient Methods of Nonlinear Optimization", ICNN'87, San Diego, CA, 1987.

[4] Fletcher, R. and Powell, M.J.D., "A Rapidly Convergent Descent Method for Minimization", *The Computer Journal,* 6:163-168, 1963.

[5] Fletcher, R., *Practical Methods of Optimization,* John Wiley & Sons, 1980.

[6] Cybenco, G.,"Approximations by Superpositions of a Sigmoidal Function", *Mathematics of Control, Signals & Systems*, vol.2, no.4, 1989.

[7] Dennis, J.E. and Schnabel, R.B., *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, New Jercey, 1983.

[8] Moré, J.J., "The Levenberg-Marquardt Algorithm: Implementation and Theory", *Numerical Analysis,* Lecture Notes in Mathematics, 630, Springer-Verlag, 1977.

[9] Yuan, J.-L. and Chiang, H.-D., "Towards Optimal Feedforward Neural Networks with Learning and Generalization Capabilities", *Proc. of First Intern'l Conf. on Appl. of Neural Networks to Power Systems,* Seattle, July 1991.

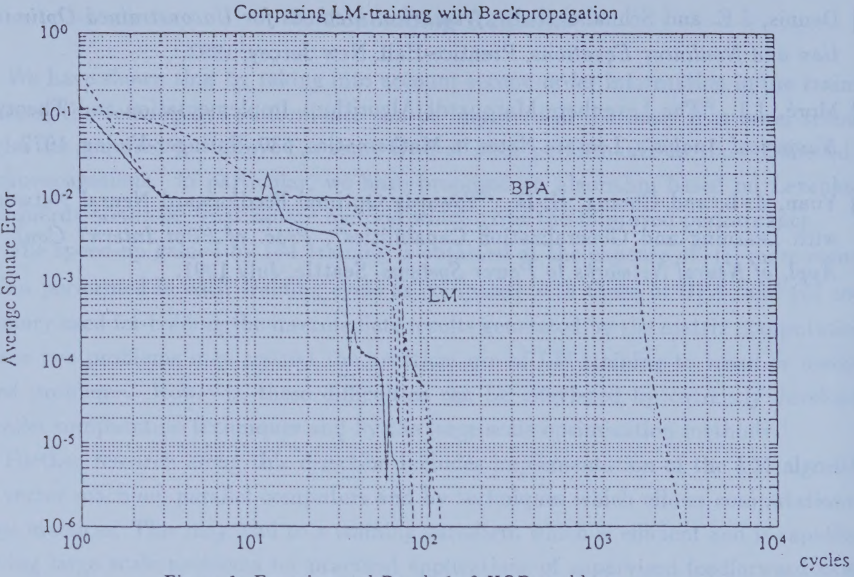


Figure 1: Experimental Result 1: 3-XOR problems.

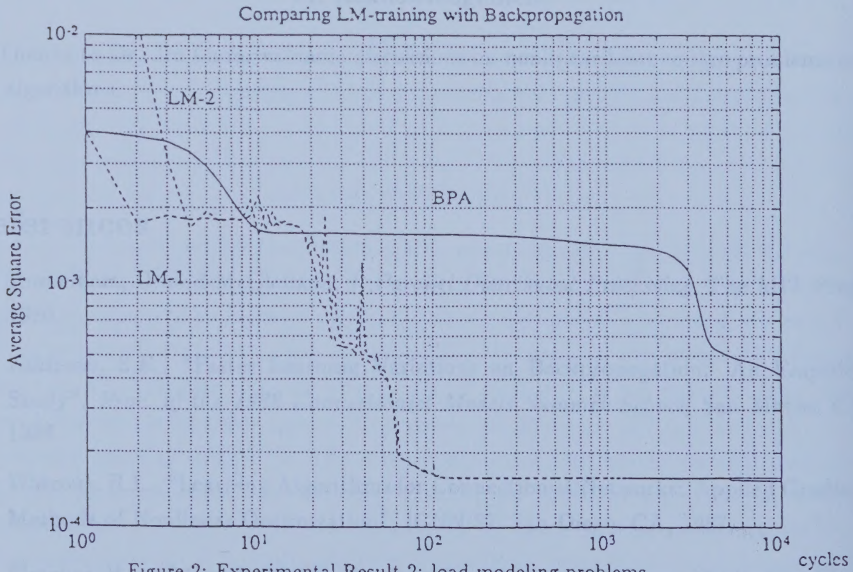


Figure 2: Experimental Result 2: load modeling problems.

A Performance Analysis Technique for Nonlinear Convolutional/Trellis-Coded Modulations

Yow-Jong Liu

Mobile Systems Department
GTE Laboratories, Waltham, MA 02254

Abstract

In this paper, a generating function techniques for analyzing bit error performance of (linear or nonlinear) convolutional-coded digital communication systems are considered. We show that the *nonlinear* feedforward convolutional encoders used with time-invariant modulation functions can be transformed into equivalent *linear* convolutional encoders with time-varying modulation functions. For this class of convolutional coded communication systems, the error performance can be analyzed with complexity $N \times N$, where N is the number of the states of the convolutional encoder, provided that conditions, discussed in this paper, are satisfied. Application of the technique to the analysis of trellis-coded modulation schemes with underlying nonlinear convolutional encoders is also considered.

1 Introduction

It has been shown that Trellis-Coded Modulation (TCM) encoders can be modeled as nonlinear systems with finite memory [1]. Through the using of orthogonal Volterra series representation, the underlying linear convolutional encoder has been identified to be rate-k/ν, where 2^k is the size of input alphabet and ν is the number of nonzero terms in the Volterra series expansion [2].

In this paper, we use an alternative approach to identify the underlying linear convolutional encoder in a digital communication system encoded by a nonlinear convolutional encoder. We show that a system with a nonlinear convolutional encoder followed by a time-invariant modulation function can be transformed into a linear convolutional encoder followed by a time-varying transmission symbol duration-dependent modulation function. This result can also be applied to TCM schemes with underlying nonlinear convolutional encoders. Based on the model described in this paper, a generating function technique for analyzing the bit error performance of the system with (linear or nonlinear) convolutional encoder is derived.

The rest of this paper is organized as follows. Section 2 describes the system model in this study. Section 3 describes the model of system with linear (or nonlinear) convolutional encoder. Section 4 contains the error performance analysis based on the generating transfer function technique. Finally the conclusions of this study are given in section 5.

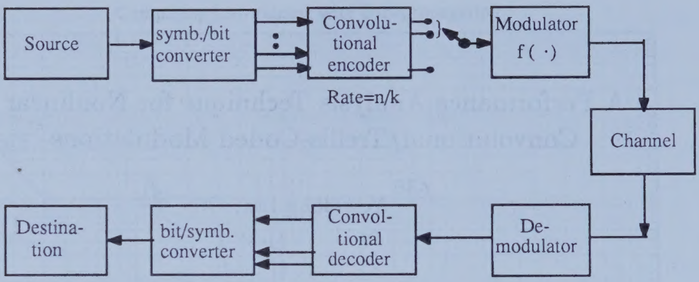


Figure 1: Block diagram of a simplified digital communication system with rate-n/k convolutional encoder.

2 System Model

Fig. 1 shows the block diagram of the end-to-end system under investigation. Consider a source symbol $\mathbf{b}_i \in \Lambda$ emitted from a discrete memoryless source (DMS) at time i, where Λ is the source alphabet with size $M = 2^k$. Each symbol $\mathbf{b}_i$ can be represented by a k-tuple binary vector

$$\mathbf{b}_i = (b_i^{(1)}, \dots, b_i^{(k)}), \tag{1}$$

where $b_i^{(l)} \in (0,1), \forall\, l = 1, \dots, k$. Each element in the n-tuple binary vector $(C_i^{(1)}, \dots, C_i^{(n)})$, which is generated by the linear (or nonlinear) convolutional encoder at time i, depends not only on the present input vector $\mathbf{b}_i$, but also on the previous μ input vectors $\{\mathbf{b}_{i-1}, \dots \mathbf{b}_{i-\mu+1}\}$, where μ is the memory length of the convolutional encoder. We can write

$$C_i^{(l)} = g_l(b_i^{(1)} \dots b_i^{(k)}, \dots, b_{i-\mu+1}^{(1)}, \dots, b_{i-\mu+1}^{(k)}), \tag{2}$$

where $C_i^{(l)} \in (0,1), \forall\, l = 1, \dots, n$ and $g_l(\cdot)$ represents all the possible operations on the input bits. In general, linear convolutional encoders contain only modulo-two sum (or Exclusive-OR) operations. In this paper, we assume that three operations: AND, OR, and Invert operations are also allowed in the case of nonlinear convolutional encoders.

Depending on the size of the channel alphabet, n-tuple binary $(C_i^{(1)}, \dots, C_i^{(n)})$ is equally divided into n' (n/n')-tuple vectors. Each vector is mapped into a channel symbol $x_{i,j} \in \Lambda'$,

$$x_{i,j} = f(C_i^{(1+(j-1)*n')}, \dots, C_i^{(j*n')}), \tag{3}$$

where $j = 1, \dots, n'$, $f(\cdot)$ is the modulation function, and Λ' is the channel input alphabet with size $2^{n/n'}$. These encoded channel symbols are then transmitted over a noisy channel.

At the receiver, the noise-corrupted encoded signal is decoded by using a maximum-likelihood sequence estimator (MLSE), such as the Viterbi algorithm. The output bits from the MLSE are then converted back to the transmitted symbols with alphabet size M.

In conventional convolutional-encoded digital communication systems, the transmission symbol duration is partitioned into n' durations. Each duration is used to transmit the coded channel symbols. In such a case, the bandwidth of the system is expanded by a factor of n'/k. We can choose $n' = n$ and $M' = 2^{k+1}$ and carefully assign channel symbols to the trellis, created by the underlying convolutional encoder, by following Ungerboeck's four basic rules [3]. It has been shown that 3-6 dB coding gain can easily be achieved in an additive white Gaussian noise channel without bandwidth expansion. Such a special design is called a trellis-coded modulation (TCM) scheme. In other words, TCM is a special case of convolutional-encoded modulation systems with a special code rate and a special channel symbol assignment to the trellis created by the underlying convolutional encoder. As a result, a general model for the convolutional-encoded modulation scheme, which we discuss in the next section, can be directly applied to TCM schemes.

3 Model for Convolutional/Trellis-Coded Modulation

First, we note that the following one-to-one relationships exist between the operations in the Boolean algebra system and the real number system.

Fact 1:

$$\begin{aligned} b_i \wedge b_j &\leftrightarrow b_i \cdot b_j, & (4)\\ b_i \oplus b_j &\leftrightarrow (1-b_i)\cdot b_j + (1-b_j)\cdot b_i, & (5)\\ b_i \vee b_j &\leftrightarrow b_i \cdot b_j + (1-b_i)\cdot b_j + (1-b_j)\cdot b_i, & (6)\\ \overline{b_i} &\leftrightarrow 1 - b_i, & (7)\end{aligned}$$

where $b_i, b_j \in \{0,1\}$, $\wedge, \vee, \oplus$, and $\overline{x}$ represent AND, OR, Exclusive OR and Invert operations in the Boolean algebra system and $\cdot, +, -$ represent the multiplication, addition, and subtraction operations. The above one-to-one correspondence simply means that all the Boolean algebraic operations can be represented by the real number operations which involve only addition, substraction, and multiplication. For the sake of brevity, we use $b_i b_j$ to represent $b_i \cdot b_j$ in the rest of the paper.

Fact 2:

$$b_1' \cdots b_h' \leftrightarrow \begin{cases} 2(b_1 \oplus \cdots \oplus b_h) - 1 & \text{if } h \text{ is odd} \\ 1 - 2(b_1 \oplus \cdots \oplus b_h) & \text{if } h \text{ is even.} \end{cases} \qquad (8)$$

where

$$b_i = \frac{b_i' + 1}{2}, \qquad (9)$$

$b_i \in \{0,1\}$, $b_i' \in \{-1,1\}$, $\forall i = 1, \ldots, h$. As we shall see later in the derivation, (8) and (9) will be useful in identifying part of the underlying linear convolutional encoder structure in the coding system.

If we rewrite (3) with explicit operations involved in the modulation function

$$x_{i,j} = f(C_i^{(1+(j-1)*n')}, \ldots, C_i^{(j*n')}; \wedge, \vee, \oplus, \overline{x}). \qquad (10)$$

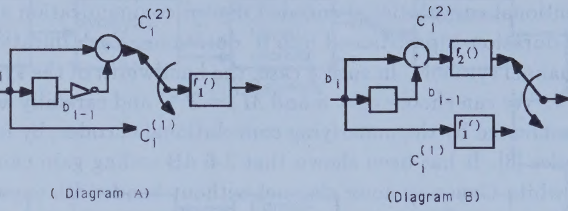


Figure 2: Diagram A: rate-1/2 nonlinear convolutional encoder with time-invariant modulation function, $f_1(\cdot)$, Diagram B: equivalent rate-1/2 linear convolutional encoder with time varying modulation function, $f_1(\cdot), f_2(\cdot)$

and substitute (2) into above equation, we have

$$x_{i,j} = f(g_{1+(j-1)*n'}(\mathbf{b}_i, \ldots, \mathbf{b}_{i-\mu+1}; \wedge, \vee, \oplus, \overline{x}), \ldots, g_{j*n'}(\mathbf{b}_i, \ldots, \mathbf{b}_{i-\mu+1}; \wedge, \vee, \oplus, \overline{x})). \quad (11)$$

From fact 1, we can replace each Boolean algebra operation in the above equation by the associated real number operation, giving

$$x_{i,j} = f(\mathcal{G}_{1+(j-1)*n'}(\mathbf{b}_i, \ldots, \mathbf{b}_{i-\mu}; +, -, \cdot), \ldots, \mathcal{G}_{j*n'}(\mathbf{b}_i, \ldots, \mathbf{b}_{i-\mu}; +, -, \cdot)). \quad (12)$$

By using (8) and (9), we can further convert function $\mathcal{G}_i$ into sum of several $\mathcal{H}$ functions which contain $b_1 \oplus \cdots \oplus b_h$ terms only. We can redefine the modulation function for each $j-th$ transmission sub-interval to be

$$x_{i,j} = f_j(\mathcal{H}_1(\mathbf{b}_i, \ldots, \mathbf{b}_{i-\mu+1}; \oplus), \ldots, \mathcal{H}_{L(j)}(\mathbf{b}_i, \ldots, \mathbf{b}_{i-\mu+1}; \oplus), \quad (13)$$

where $L(j)$ depends on the total number of $b_1 \oplus \cdots \oplus b_\ell$ type terms in the function $\mathcal{G}_\ell, \forall\ \ell = 1 + (j-1) * n', \ldots, j * n'$. If we compare (3) with (13), we observe that the equivalent underlying rate-k/ν linear convolutional encoder is

$$c_i^{(l)} = \mathcal{H}_l(b_i^{(1)}, \ldots, b_i^{(k)}, \ldots, b_{i-\mu+1}^{(1)}, \ldots, b_{i-\mu+1}^{(k)}), \quad (14)$$

where $l = 1, \ldots, \nu$ and

$$\nu = \sum_{j=1}^{n'} L(j). \quad (15)$$

The modulation function is time-varying and depends on the specific transmission symbol interval.

Example 1. Fig. 2 shows a rate-1/2 nonlinear convolutional encoder followed by a fixed modulation function $f_1(\cdot)$. If we assume that BPSK modulation is used, the output symbols are

$$\begin{aligned} x_{i,j} &= f_1(C_i^{(j)}) && (16) \\ &= e^{jC_i^{(j)}\pi}, \qquad j = 1, 2 && (17) \end{aligned}$$

where

$$C_i^{(1)} = b_i \quad (18)$$

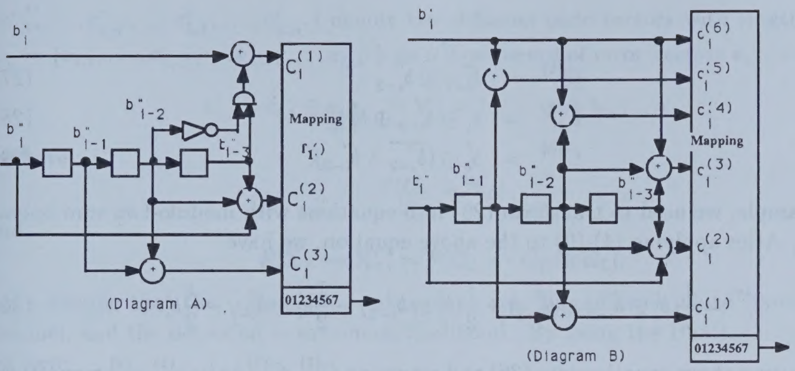


Figure 3: 8-state phase-invariant trellis-coded modulation: Diagram A: rate-2/3 non-linear convolutional encoder with modulation function, $f_1(\cdot)$, Diagram B: equivalent rate-2/6 linear convolutional encoder with different modulation function, $f_2(\cdot)$

and

$$C_i^{(2)} = b_i \oplus \overline{b_{i-1}}. \tag{19}$$

We observe that (18), which is equal to the input bit, can be part of the linear convolutional encoder. To find the remainder of the equivalent linear convolutional encoder, we need only modify (19). By applying (7)-(9), we have

$$C_i^{(2)} = 1 - b_i \oplus b_{i-1} \tag{20}$$

Substitute above equation into (17), we can redefine the new modulation function for $x_{i,2}$ as:

$$\begin{aligned} x_{i,2} &= f_2(c_i^{(2)}) && (21)\\ &= e^{j(1-c_i^{(2)})\pi}. && (22)\end{aligned}$$

The complete underlying linear convolutional encoder is

$$\begin{aligned} c_i^{(1)} &= b_i && (23)\\ c_i^{(2)} &= b_i \oplus b_{i-1} && (24)\end{aligned}$$

Diagram B in Fig. 2 shows the equivalent model with a rate-1/2 linear convolutional encoder and time varying modulation functions, $f_1(\cdot)$ and $f_2(\cdot)$. In this example $L(1) = 1$ and $L(2) = 1$.

Example 2. Consider a 90° phase-invariant TCM scheme [4] with 8-PSK modulation function as shown in Fig. 3, we can write

$$\begin{aligned} x_i &= f_1(C_i^{(1)}, C_i^{(2)}, C_i^{(3)}) && (25)\\ &= e^{j(C_i^{(1)}+2C_i^{(2)}+4C_i^{(3)})}, && (26)\end{aligned}$$

where

$$\begin{aligned} C_i^{(1)} &= b''_{i-1} \oplus b''_{i-2} && (27) \\ C_i^{(2)} &= b'_i \oplus b''_{i-2} \oplus b''_{i-3} && (28) \\ C_i^{(3)} &= b'_i \oplus (\overline{b''_{i-2} \wedge b''_{i-3}}). && (29) \end{aligned}$$

In this example, we need to transform (29) into equations with modulo-two sum operations only. After applying (4)-(9) to the above equation, we have

$$C_i^{(3)} = \frac{1}{2}(b'_i \oplus b''_{i-2} \oplus b''_{i-3}) + \frac{1}{2}(b'_i \oplus b''_{i-3}) - \frac{1}{2}b'_i \oplus b''_{i-2} - \frac{1}{2}b'_i. \tag{30}$$

By substituting above equation into (26) and renaming $C_i^{(1)}, C_i^{(2)}$ to be $c_i^{(1)}, c_i^{(2)}$,respectively, we have

$$\begin{aligned} x_i &= f_2(c_i^{(1)}, c_i^{(2)}, c_i^{(3)}, c_i^{(4)}, c_i^{(5)}, c_i^{(6)}) && (31) \\ &= e^{j(c_i^{(1)}+2c_i^{(2)}+2c_i^{(3)}+2c_i^{(4)}-2c_i^{(5)}-2c_i^{(6)})\pi}. && (32) \end{aligned}$$

The complete underlying linear convolutional encoder is defined to be

$$\begin{aligned} c_i^{(1)} &= b''_{i-1} \oplus b''_{i-2} \\ c_i^{(2)} &= b'_i \oplus b''_{i-2} \oplus b''_{i-3} \\ c_i^{(3)} &= b'_i \oplus b''_{i-2} \oplus b''_{i-3} \\ c_i^{(4)} &= b'_i \oplus b''_{i-3} \\ c_i^{(5)} &= b'_i \oplus b''_{i-2} \\ c_i^{(6)} &= b'_i \end{aligned}$$

The equivalent system with a rate-2/6 linear convolutional encoder and mapping function $f_2(\cdot)$ is shown in the diagram B in Fig. 3. It should be noted that there is only one transmission symbol interval in this example with $L(1)$ equal to 6.

4 Error probability: upper bound

Let $\mathbf{X}_L = \{x_{1,1}, \ldots, x_{1,n'}, \ldots x_{L,1}, \ldots, x_{L,n'}\}$ and $\mathbf{X}'_L = \{x'_{1,1}, \ldots, x'_{1,n'}, \ldots x'_{L,1}, \ldots, x'_{L,n'}\}$ denote two sequences of x-symbols with length $n'L$. The union bound provides the following inequality for the probability of an error event:

$$P(e) \leq \sum_{L=1}^{\infty} \sum_{\mathbf{X}_L} \sum_{\mathbf{X}'_L \neq \mathbf{X}_L} P[\mathbf{X}_L] P[\mathbf{X}_L \rightarrow \mathbf{X}'_L] \tag{33}$$

where $P[\mathbf{X}_L \rightarrow \mathbf{X}'_L]$ denotes the "pairwise" error probability, i.e., the probability that when $\mathbf{X}_L$ is transmitted $\mathbf{X}'_L$ be preferred to $\mathbf{X}_L$ by the demodulator.

From (13) we know that there exists a one-to-one correspondence between output symbols $x_{i,j}$ and $\mathbf{c}_{i,j}$-vectors. Let $\mathbf{C}_L = \{\mathbf{c}_{1,1}, \ldots, \mathbf{c}_{1,n'}, \ldots \mathbf{c}_{L,1}, \ldots, \mathbf{c}_{L,n'}\}$ and $\mathbf{C}'_L =$

$\{\mathbf{c}'_{1,1},\ldots,\mathbf{c}'_{1,n'},\ldots\mathbf{c}'_{L,1},\ldots,\mathbf{c}'_{L,n'}\}$ denote two different code vectors with length $n'L$ and $\mathbf{E}_L = \{\mathbf{e}_{1,1},\ldots,\mathbf{e}_{1,n'},\ldots\mathbf{e}_{L,1},\ldots,\mathbf{e}_{L,n'}\}$ an $n'L$-sequence of error vectors $\mathbf{e}_{i,j}$, where

$$\mathbf{c}'_{i,j} = \mathbf{c}_{i,j} \oplus \mathbf{e}_{i,j} \qquad \forall\, i = 1,\ldots \quad j = 1,\ldots,n', \tag{34}$$

we have

$$P[\mathbf{C}_L = \mathbf{X}_L] \tag{35}$$

and

$$P[\mathbf{X}_L \rightarrow \mathbf{X}'_L] = P[\mathbf{C}_L \rightarrow \mathbf{C}_L \oplus \mathbf{E}_L]. \tag{36}$$

Let's assume that the transmission takes place over the additive white Gaussian noise channel, and the detection is maximum-likelihood. By using the Bhattacharyya bound we have

$$\begin{aligned} P[\mathbf{C}_L \rightarrow \mathbf{C}_L \oplus \mathbf{E}_L] &\le \exp\left\{-\frac{1}{4N_0} \parallel f(\mathbf{C}_L) - f(\mathbf{C}_L \oplus \mathbf{E}_L) \parallel^2\right\} \\ &= \exp\left\{-\frac{1}{4N_0}\sum_{i=1}^{L}\sum_{j=1}^{n'} \parallel f_j(\mathbf{c}_{i,j}) - f(\mathbf{c}_{i,j} \oplus \mathbf{e}_{i,j}) \parallel^2\right\}. \end{aligned} \tag{37}$$

If we assume that the source symbols have equal probabilities $1/M = 2^{-k}$, we can define the $N \times N$ "error weight matrices" $G(\mathbf{e}_i)$ as follows, where

$$\mathbf{e}_i = (\mathbf{e}_{i,1} \ldots \mathbf{e}_{i,n'}) \tag{38}$$

The entry p, q of $G(\mathbf{e}_i)$ is zero if no transition from the code trellis state p to the state q is possible. Otherwise, it is given by

$$[G(\mathbf{e}_i)]_{p,q} = \frac{1}{M}\sum_{\mathbf{c}_{p-q}} Z^{\sum_{j=1}^{n'} \| f_j(\mathbf{c}_{j,p-q}) - f_j(\mathbf{c}_{j,p-q} \oplus \mathbf{e}_{i,j}) \|^2} \tag{39}$$

where $Z = \exp(-1/4N_0)$ and $\mathbf{c}_{j,p\rightarrow q}$ are the vectors generated by the transition from p to q during the j-th subinterval of a channel symbol duration. (The sum accounts for the possible parallel transitions.)

With these notations, to any vector $\mathbf{E}_L$ there corresponds a sequence of L error weight matrices $G(\mathbf{e}_1), \cdots, G(\mathbf{e}_L)$, and we have

$$G(\mathbf{E}_L) = \frac{1}{N}\mathbf{1}'\prod_{n=1}^{L} G(\mathbf{e}_n)\mathbf{1}, \tag{40}$$

where $\mathbf{1}$ is the column N-vector all of whose elements are 1. From (35), (36), (33), (37), and (40) , we can show that the error event probability is

$$\begin{aligned} P(e) &\le T(Z) && (41) \\ &= \frac{1}{N}\mathbf{1}'G\mathbf{1}, && (42) \end{aligned}$$

where the matrix

$$G = \sum_{L=1}^{\infty}\sum_{\mathbf{E}_L \neq 0}\prod_{i=1}^{L} G(\mathbf{e}_i).$$

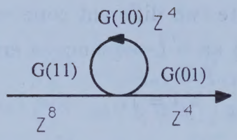


Figure 4: Error state diagram for Example 1.

The bit error probability can also be derived by modifying the error matrices. Each element in the matrix is multiplied by a factor I^{ϵ}, where ϵ is the number of incorrect input bits associated with the error matrices. The upper bound for the bit error is then:

$$P_b \leq \frac{1}{k}\frac{d}{dI}T(Z,I)\bigg|_{I=1},$$

where k is the number of source bits per trellis transition. Another upper bound (TUB) can be obtained by substituting for the Bhattacharyya bound in (37) a tighter inequality [5]:

$$P_b \leq \frac{1}{m}Q(\frac{d_{\text{free}}}{\sqrt{2N_0}})\exp(d_{\text{free}}^2/4N_0)\frac{d}{dI}T(Z,I)\bigg|_{I=1}.$$

It should be noted that the sequence of error vectors is not made of independent elements. The connections between error vectors $\mathbf{e}_i$ can be described by a state diagram similar to the one describing the code, and which we shall call the *error state diagram*, as distinguished from the code state diagram. The error state diagram has a structure determined by the linear convolutional code in our model, and differs from it only in the denomination of its states and the branch labels, which are the matrices $G(\mathbf{e}_i)$.

If $\mathbf{1}$ is either an eigenvector of all the matrices $G(\mathbf{e})$ or its transpose $G'(\mathbf{e})$, i.e.,

$$\mathbf{1}'G(\mathbf{e}) = K_{G(\mathbf{e})}\mathbf{1}',$$

or

$$G'(\mathbf{e})\mathbf{1} = K'_{G(\mathbf{e})}\mathbf{1},$$

where $K_{G(\mathbf{e})}$ and $K'_{G(\mathbf{e})}$ are some constants, then the transfer function can be computed by using scalar labels on the branches of the error-state diagram. These labels are the sums of the elements in a row (column). As defined in [2], if any one of the conditions hold then we say that the convolutional-encoded modulation scheme is uniform.

Example 1 (continued). The group of error vectors is

$$00, 11, 01, 10.$$

The error weight matrices corresponding to nonzero errors are

$$G(11) = \frac{1}{2}\begin{bmatrix} Z^8 & Z^8 \\ Z^8 & Z^8 \end{bmatrix}$$

$$G(10) = \frac{1}{2}\begin{bmatrix} Z^4 & Z^4 \\ Z^4 & Z^4 \end{bmatrix}$$

$$G(01) = \frac{1}{2}\begin{bmatrix} Z^4 & Z^4 \\ Z^4 & Z^4 \end{bmatrix}.$$

The corresponding error state diagram is shown in Fig. 4. We can easily observe that this nonlinear convolutional-coded modulation is uniform. The matrix G is given by

$$G = G(11)\,[I_N - G(10)]^{-1}\,G(01), \tag{43}$$

which yields

$$G = \frac{1}{2}\frac{Z^{12}}{1-Z^4}\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \tag{44}$$

and the transfer function is given by

$$T(Z) = \frac{Z^{12}}{1-Z^4}. \tag{45}$$

Since this coded-modulation scheme is uniform, we can use the equivalent scalar branch labels instead of the matrix branch labels in the error state diagram. Fig. 4 also shows the equivalent scalar branch labels. We can easily show that the transfer function is equal to (45). The upper bound of the bit error probability is given by

$$P_b \leq Q(\sqrt{\frac{6}{N_o}})\frac{Z^{24}}{(1-Z^4)^2}.$$

Example 2 (continued). The group of error vectors is

$$000000, 010011, 110101, 100110, 100000, 110011, 010101, 000110,$$

$$011111, 001100, 101010, 111000, 111111, 101100, 001010, 011001.$$

It can be shown that this code is uniform. Fig. 5 shows the simulated performance and calculated upper bound.

5 Conclusion

We have shown that the underlying *nonlinear* convolutional encoder and time-invariant modulation function can be transformed into an equivalent *linear* convolutional encoder with transmission symbol duration-dependent time-varying modulation functions. For these type of convolutional-coded communication systems, the error performance can be analyzed with complexity $N \times N$, where N is the number of the state of the convolutional encoder, provided that the conditions discussed in this paper are satisfied. Application of the technique to the analysis of trellis-coded modulation schemes with underlying nonlinear convolutional encoder are also considered.

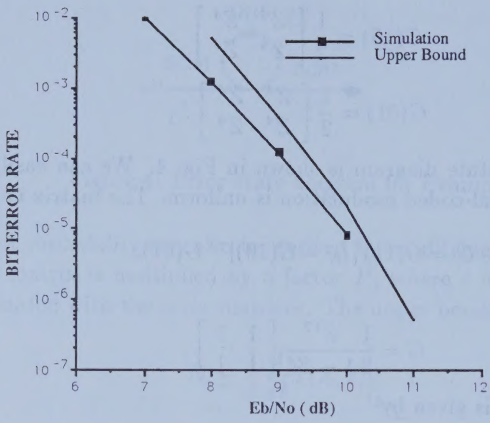


Figure 5: Performance of 8-state TCM scheme over AWGN channel

References

[1] A. R. Calderbank and J. E. Mazo, "A new description of trellis codes," *IEEE Trans. Inform. Theory,* vol. IT-30, pp. 784-791, Nov. 1984

[2] Y.-J. Liu,I. Oka and E. Biglieri, "Error probability for digital transmission over nonlinear channels with application to TCM," *IEEE Trans. Inform. Th.* Vol. 36, No. 5, Sept. 1990

[3] G. Ungerboeck, "Channel Coding With Multilevel/Phase Signals," *IEEE Trans. Inform. Th.*, vol. IT-28, pp. 56-67, January 1982

[4] Z. C. Zhu and A. P. Clark, "Rotationally invariant coded PSK signals," *IEE Proceedings,* Vol.134, Pt. F, No.1, February 1987

[5] S. Benedetto, M. Ajmone Marsan, G. Albertengo, and E. Giachin, "Combined coding and modulation: Theory and applications," *IEEE Trans. Inform. Theory.* Vol. IT-34, No. 2, pp. 223-236, March 1988

Optimum Block Based Signaling over $1\pm D$ and $1-D^2$ Partial Response Channels

A. K. Khandani[1] and P. Kabal[1,2]

Dept. of Elec. Eng., McGill University, 3480 University St., Montreal, Canada, H3A-2A7
INRS-Telecom, 3 Place du Commerce, Ile-des-Soeurs, Verdun, Canada, H3H-1H6

Abstract: In this work, we discuss the optimum block based signaling over partial response channels. This is composed of selecting a basis for the space (modulating waveforms) and a boundary (shaping region) for the constellation. In specific, channels $1\pm D$ and $1-D^2$ are studied. It is shown that the optimum basis for these channels is either sine basis or closely related to it. This facilitates the modulation operation by the use of the fast sine transform algorithms. Selection of the constellation (including its dimensionality) is based on minimizing the loss with respect to a flat channel. This allocates equal energy to the nonempty dimensions. Optimum basis is compared with the Fourier basis. Capacity of these channels is computed. We also calculate the power spectrum associated with the proposed schemes.

1. Introduction

Any channel with a nonflat transfer function has memory. This memory results in intersymbol interference (ISI) between successive transmissions. The ISI acts as a source of impairment. Partial response channels are nonflat channels with a linear Finite-Impulse-Response (FIR) transfer function. There are two main approaches for dealing with the problem of ISI in the partial response channels.

In the first approach, one tries to undo the memory. This is based on cascading the channel with its inverse. Inverse filter is implemented in either the receiver or the transmitter side. Precoding methods are examples of the transmitter filter. In this case, the addition operation is defined such that the set of constellation points form a group under that addition. In the case of the Voronoi constellations, this is achieved by performing the addition modulo the shaping lattice, [1]. Similar modulo operation is achieved at the receiver side. This changes the ordinary addition of the (FIR) channel to the desired modulo form. Conventional addition modulo K in an $2K$ level PAM signal is the special case corresponding to a cubic constellation. Inverse filter can be also implemented in the receiver side. This is achieved by subtracting the effect of the previously sent symbols from the received signal. This results in a memoryless channel but with colored noise. Decision Feedback Equalizer (DFE) is an example of this approach.

In the second approach, one considers the effect of the channel memory as part of the encoding process. This method usually requires a decoder with a large number of states.

We assume that the channel has a memorylength of M_0 symbols (impulse response of length M_0+1) and coding is achieved over N dimensional blocks. In one approach, each block starts from zero initial conditions. This is achieved by sending M_0 zeros between successive blocks. In this case, the ISI is confined to the symbols within a block. This is obtained at the price of lossing M_0 dimensions per each block. For large values of N, the loss in dimensionality is small. In another approach, no idle time is inserted between successive blocks. This results in no loss of dimensionality but has its own problems. For example, in this case the channel inverse is Infinite-Impulse-Response (IIR). This may result in a system subject to instability or to a system with highly correlated noise at the receiver side. This colored noise is the cause of the error accumulation in the DFE.

In all these methods, some dimensions are lost at the transmitter side and/or some useful information are discarded at the receiver side. This results in a suboptimum system. An optimum scheme should work on the whole transmission period as a single block and achieve the encoding/decoding process on the whole block.

In the following, we introduce the block diagram of the system under consideration.

2. System block diagram

The block diagram of the system is shown in Fig. 1. The channel has a memorylength of M_0 symbols. The noise is additive white Gaussian with zero mean and unity power. Each signaling interval (block) invokes M channel uses. The last M_0 transmissions of each block are zero. As a result, at the beginning of each $N{=}M-M_0$ dimensional block, channel is in zero state. From the available N dimensions, $N_0 \leq N$ are used for transmission and $N-N_0$ dimensions remain empty.

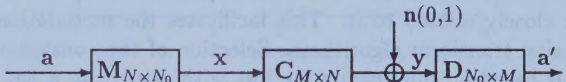


Fig. 1 System block diagram.

In each signaling interval, the data bits select a point $\mathbf{a}$ within the input constellation. This is a set of N_0 dimensional points bounded within a shaping region. We assume that the constellation points are used with equal probability. The $N \times N_0$ orthonormal matrix $\mathbf{M}$ is the basis of the constellation. Using the orthonormality of $\mathbf{M}$, one can show that: (i) the average energy and the rate of $\mathbf{x}$ and $\mathbf{a}$ are the same, (ii) for $N{=}N_0$, if $\mathbf{a}$ is white and its components are of equal energy, $\mathbf{x}$ will be also white and its components will be of equal energy. It can be also shown that: (i) if the constellation is closed under sign changings of the coordinates, $\mathbf{a}$ will be white, (ii) if the constellation is closed under the permutation of the coordinates, the components of $\mathbf{a}$ will have equal energy.

The $M \times N$ matrix $\mathbf{C}$, is the transfer matrix of the channel. For a memoryless channel, $\mathbf{C}$ is diagonal. At the channel output, we receive the M dimensional vector $\mathbf{y}$. The $N_0 \times M$ demodulator matrix $\mathbf{D}$ is selected such that $\mathbf{DCM} = \mathbf{I}$, $\mathbf{I}$ is the $N_0 \times N_0$ identity matrix. Matrix $\mathbf{D}$ also affects the statistics of the noise. In general, as the decoding operation is linear, the noise remains Gaussian. In our case (white Gaussian noise with unity power), the whole system is equivalent a flat channel with an additive Gaussian noise of autocorrelation $\mathbf{DD}^t$. The noise will be white if $\mathbf{DD}^t$ is diagonal. For a colored noise, decisions along different dimensions are not independent of each other. The optimum decision in this case should be based on maximizing the likelihood in the corresponding N_0 dimensional space. We assume a suboptimum method in which the decision along each dimension is made independently. In this case, the noise power along the i'th dimension, $\sigma_i^2, i = 0, \ldots, N_0 - 1$, is equal to the i'th diagonal element of $\mathbf{DD}^t$ (i.e., equal to the square norm of the i'th row of $\mathbf{D}$).

Selection of the constellation is composed of three different parts. The first part is the selection of $\mathbf{M}$. The other two parts, like over the falt channel, are constellation shaping and channel coding. Shaping concerns the selection of the constellation boundary at the channel input. Channel coding concerns the minimum distance to noise ratio at the channel output. The objective is to minimize the probability of error between the constellation points for a given total rate and energy.

In continuous approximation, the distribution of the constellation points is approximated by a continuous uniform density within the shaping region. Assuming continuous approximation, one can show that shaping, channel coding and modulation can be selected independently. In this case, shaping and channel coding are similar to the case of a flat channel. It should be mentioned that in the discrete analysis, shaping and coding are not independent and one gains by using a 'combined coding and shaping' scheme, [2].

In shaping the boundary of a constellation, one tries to minimize the average energy of the set for a given number of points. In our case, assuming continuous approximation, shaping is achieved by selecting an N_0 dimensional solid which has the least second moment for a given volume. The optimum shaping regions, introduced in [3], have equal second moments along different dimensions. These regions result in a white PAM vector $\mathbf{a}$ which has the maximum possible rate subject to: (i) a constraint on the total average energy, (ii) a constraint on the shaping redundancy. Using an optimum shaping region results in equal distribution of energy between the nonempty dimensions. In this case, the dimensions are used in an on-off manner and a dimension is either empty or has the same energy as other dimensions. In the proposed method, the selection of the nonempty dimensions is based on minimizing the loss with respect to a flat channel.

To obtain a minimum distance to noise ratio of one at the demodulator output, the minimum distance along the i'th dimension of the input constellation should be equal to σ_i. This is obtained by scaling the coding lattice with a factor proportional to σ_i along the i'th dimension. In this case, the volume of the Voronoi region at the channel input is proportional to $\prod_i \sigma_i$. For a fixed number of points (rate), to minimize the volume of the constellation and consequently its average energy, one should select the basis to minimize $\prod_i \sigma_i$. For the case that all the dimensions are nonempty, it is shown in [4] that the optimum basis is the input eigenvectors of $\mathbf{C}$, i.e., eigenvectors of $\mathbf{C}^t\mathbf{C}$. With the same approach as in [4], it can be shown that if some of the dimensions are empty, the optimum basis is composed of the input eigenvectors of the channel corresponding to the largest eigenvalues. In this case, $\mathbf{D}\mathbf{D}^t$ is diagonal and the noise along different dimensions are uncorrelated (independent).

To reduce the computational complexity, the optimum basis is usually replaced by the Fourier basis, i.e.,

$$\mathbf{M} = \frac{1}{\sqrt{N}}\left[M(m,n) = \exp\left(\frac{2\pi mn}{N}\right)\right], \qquad (1)$$

where $m, n{=}0, \ldots, N{-}1$.

In the following, we study the channels $1 \pm D$ and $1-D^2$ in more details. These are three important examples of the partial response channels, [5].

3. Eigenvectors/Eigenvalues of $1 \pm D$ and $1-D^2$ channels

For $1 \pm D$ channels, $M{=}N{+}1$ and the i'th column of $\mathbf{C}$, $i{=}0, \ldots, N{-}1$, is equal to, $[(0)^i, \sqrt{2}/2, \pm\sqrt{2}/2, (0)^{N-1-i}]$. For $1-D^2$ channel, $M{=}N{+}2$ and the i'th column of $\mathbf{C}$, $i{=}0, \ldots, N{-}1$, is equal to, $[(0)^i, \sqrt{2}/2, 0, -\sqrt{2}/2, (0)^{N-1-i}]$. For $1-D$ channel, the eigenvectors are sine function, i.e.,

$$\mathbf{M} = \left[M(m,n) = \sqrt{\frac{2}{N+1}}\ \sin\ \frac{\pi(m+1)(n+1)}{N+1}\right], \qquad (2)$$

where $m, n{=}0, \ldots, N{-}1$. The corresponding eigenvalues are equal to,

$$\phi_m^2 = 1 - \cos\frac{\pi(m+1)}{(N+1)}\,. \tag{3}$$

To prove this, we consider $\mathbf{C}^t\mathbf{C}$ as the transform matrix of a linear time invariant system with transfer function $H(D) = 0.5\,(1{-}D)(1{-}D^{-1})$. We apply causal input and truncate the output to positive time. In this case, if $M(n)$ is an input eigenvector, we should have,

$$H(D)\left[\bar{M}(D) - M(0)\right] + M(0)(1 - 0.5\,D) = \phi^2\bar{M}(D)\,, \tag{4}$$

where $\bar{M}(D)$ is the transform of $M(n)$ and ϕ^2 is the corresponding eigenvalue. Calculating (4) at time zero results in,

$$\phi^2 = 1 - 0.5\frac{M(1)}{M(0)}\,. \tag{5}$$

Substituting (5) in (4), we obtain,

$$\bar{M}(D) = \frac{M(0)}{1 - \dfrac{M(1)}{M(0)}D + D^2}\,. \tag{6}$$

Equations (6) and (5) are satisfied by the eigenvectors and eigenvalues given in (2) and (3).

This can be also verified by considering Eq. (2) as a periodic function with period $N{+}1$. This function is zero at $n = i(N{+}1){-}1$, $\forall i$. This means that the periodic function itself provides zero initial condition for successive N dimensional blocks. Consequently, the response of the system in each block starts from the steady state. On the other hand, in steady state a sinosoid is the input eigenfunction of any linear system.

The eigenvectors of $1{+}D$ channel are obtained by multiplying (2) by $(-1)^n$. The eigenvalues are given in Eq. 3.

The eigenvalues of $1{-}D^2$ channel, N even, are in pair equal to the eigenvalues of an $N/2$ dimensional $1{-}D$ channel. The eigenvectors are obtained by time multiplexing the eigenvectors of the corresponding $1{-}D$ channel by its negative or by zero and normalizing the power to unity (or more generally by any linear combination of these two cases with the power normalized to unity). For all the three channels, the multiplication by $\mathbf{M}$ can be performed by using the fast sine transform algorithms.

We also have, $\prod_i \phi_i^2 = |\mathbf{C}^t\mathbf{C}|$ where $|\mathbf{C}^t\mathbf{C}|$ is the determinant of $\mathbf{C}^t\mathbf{C}$. For $1{\pm}D$ channels this is equal to,

$$\prod_{i=0}^{N-1}\phi_i^2 = 2^{-N}(1+N)\,. \tag{7}$$

For $1{-}D^2$ channel, N even, this is equal to,

$$\prod_{i=0}^{N-1}\phi_i^2 = 2^{-N}(1+0.5N)^2\,. \tag{8}$$

In general, using the input eigenvectors for modulation results in $\sigma_i = 1/\phi_i$.

4. Capacity

The capacity of the system under consideration is equal to the sum of the capacities along different dimensions. This is due to the fact that the decision along each dimension is made independently. In this case, to maximize the sum of the capacities, the energy is distributed such that the sum of the signal and noise power along all the dimensions is the same, [6]. This is known as the water filling method. This method can be justified by considering the sphere hardening argument, [7]. In this case, the normalized signal space at the channel output is an ellipsoid. The length of the i'th axis of this ellipsoid is determined by the sum of the signal and noise power along the i'th dimension. For a fixed total energy, the volume is maximized if the length of all the axises are equal. This results in a spherical region. In this case, the entropy of the channel output (entropy of the signal plus noise) along all the dimensions is the same. This is due to the one-to-one relationship between the entropy and energy of a Gaussian process. This procedure may allocate zero energy to some of the dimensions.

It can be shown that the basis which maximizes the capacity is composed of the eigenvectors of $\mathbf{C}^t\mathbf{C}$ corresponding to the largest eigenvalues. For large M, the effect of the initial conditions is negligible and the eigenfunctions tend to complex exponentials. The corresponding eigenvalues are equal to the power spectrum of the channel impulse response at the corresponding frequency.

Mathematically, the capacity per channel use is calculated from,

$$C = \frac{1}{2M} \sum_{i:\sigma_i^2 \leq B} \log_2 \left(\frac{B}{\sigma_i^2} \right), \tag{9}$$

where $B{=}E_i{+}\sigma_i^2$ is selected such that $\sum_i E_i{=}ME$, E_i is the energy allocated to i'th dimension and E is the available energy per channel use. For the case that the eigenvectors are used as the basis, for $1{\pm}D$ channel, using (3), we obtain,

$$B = \frac{1}{N_0} \left\{ ME + \frac{1}{2} \sum_{i=M-N_0-1}^{M-2} \sin^{-2} \left[\frac{\pi(i+1)}{2M} \right] \right\} < \frac{1}{2} \cos^{-2} \left[\frac{\pi(N_0+1)}{2M} \right], \tag{10}$$

where N_0, number of nonempty dimensions, is the largest integer in the range $[1, M{-}1]$ satisfying the right hand side inequality. The loss with respect to a flat channel is the ratio of the energy per channel use of the two systems when the capacities are the same. Figure 2 shows this loss for different dimensionalities.

An M dimensional, M even, $1{-}D^2$ channel can be considered as two time multiplexed $M/2$ dimensional $1{-}D$ channels. This means that the capacity of this channel is equal to two times the capacity of an $M/2$ dimensional $1{-}D$ channel.

For large E, all the dimensions are nonempty, i.e., $N_0{=}N{=}$. In this case, assuming $E_i \simeq B$ and using Eqs. (7) and (8) in (9), it is easy to verify that the capacity of all the three channels is 0.5 bits per dimension less than the capacity of a flat channel with the same additive noise. This corresponds to 3 dB loss of energy.

The water filling method may seem to be in contradiction with the argument of allocating equal energy to the nonempty dimensions. The contradiction is explained by considering that:

- For a Gaussian channel, capacity is achieved by a Gaussian source and for such a source there exists a one-to-one correspondence between the energy and the entropy.

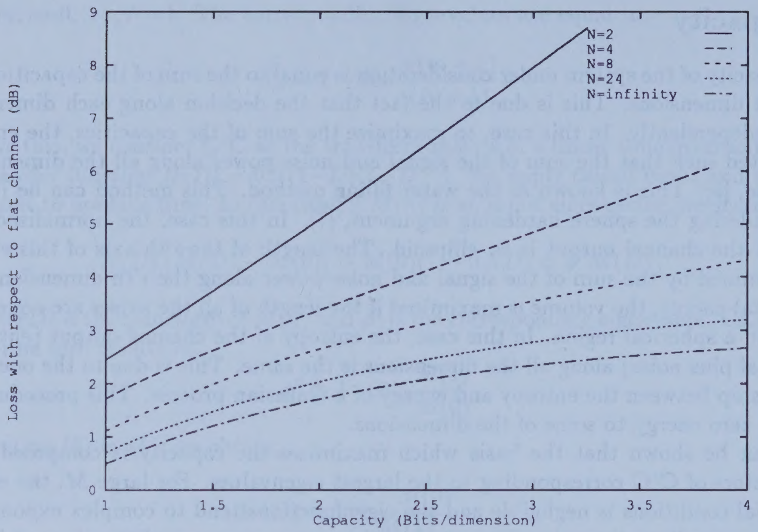


Fig. 2 Loss of energy in 1±D channel, capacity analysis.

This means that in the capacity analysis, rate and energy can not be distributed independently. On the other hand, in a finite dimensional coding scheme, these two factors are independent.

- Capacity is determined by the volume of the signal space at the channel output. This volume is maximized by applying the water filling method. In a finite dimensional coding scheme, the volume of the signal space at the channel input determines the rate. This volume is maximized by allocating equal energy to the nonempty dimensions.

5. Modulation

Modulation concerns the selection of the basis **M**. We assume that the channel coding and shaping are not present. This means that the points of the output constellation belong to the lattice $\mathbf{Z}^{N_0} + (1/2)^{N_0}$ and the boundary of the input constellation is a hypercube. This can be easily generalized to other cases. The channel is normalized to unity power gain. The performance is compared to a unity gain flat channel with the same additive noise. The rate to be transmitted per channel use is equal to, $R_f = \log_2(T_f)$. For a minimum distance of one, the required volume per M dimensions of the flat channel is equal to, $V_f = (T_f)^M$. Using continuous approximation, this corresponds to an average energy per dimension of $(T_f)^2/12$, [8].

For a minimum distance of one at the channel output, the volume of the Voronoi region at the channel input should be equal to $\prod_{i=0}^{N_0-1} \sigma_i$. In this case, if the edge length of the shaping hypercube is equal to T_0, we have,

$$(T_0)^{N_0} = (T_f)^M \times \prod_{i=0}^{N_0-1} \sigma_i \,. \tag{11}$$

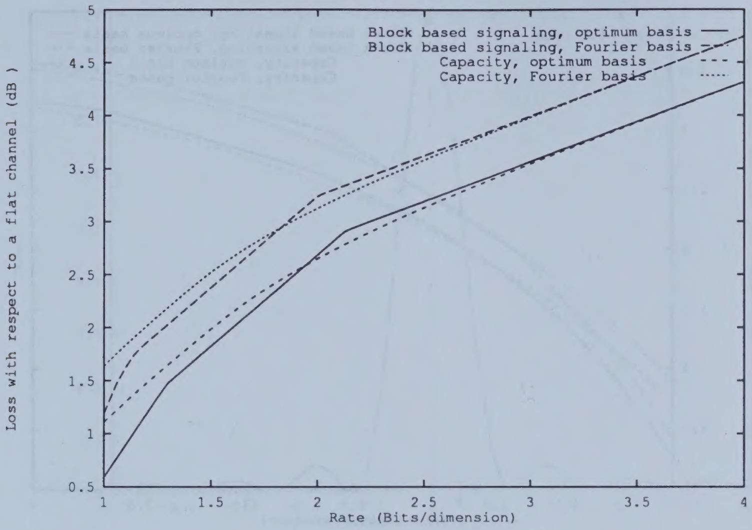


Fig. 3 Loss of energy in 1±D channel, $N=8$.

The total average energy of the coded scheme and the reference scheme are equal to, $E_0 = N_0(T_0)^2/12$ and $E_f = M(T_f)^2/12$, respectively. Using these values and (11), the loss in energy with respect to the reference system is found as,

$$10\log_{10}(\frac{E_0}{E_f}) = 6\,R_f\left(\frac{M}{N_0}-1\right) + \frac{10}{N_0}\sum_{i=0}^{N_0-1}\log_{10}(\sigma_i^2) - 10\log_{10}\left(\frac{M}{N_0}\right). \qquad (12)$$

To optimize the selection of the empty dimensions, (12) is minimized over $N_0 \leq N$ where $N = M - M_0$. The result of these calculations for $N = 8, 24$ and channels 1±D is shown in Figs. 3 and 4. Different slopes on the curves correspond to different number of empty dimensions. It is also shown the loss of the energy obtained by equating the capacities.

For large values of R_f, all the dimensions are nonempty, i.e., $N_0 = N$. In this case, substituting Eqs. (7) and (8) in (12) and tending $N \rightarrow \infty$, the asymptotic value of loss for the channels under consideration is found to be equal to 3 dB.

In the next section, we calculate the power spectrum at different points of the transmission system for the 1±D channels. Total energy per signaling interval is normalized to one. Optimumm basis (input eigenvectors of the channel) are used for modulation. It can be shown that the corresponding demodulator for the 1±D channel is qual to,

$$\mathbf{D} = \left[D(m,n) = (\pm 1)^n\sqrt{\frac{1}{N+1}}\left(\sin\frac{\pi(m+1)(n+1)}{N+1} - \sin\frac{\pi(m+1)n}{N+1}\right)\right], \qquad (13)$$

where $m, n = 0, \ldots, N-1$. We also assume that the dimensions are all nonempty, i.e., $N_0 = N = M - 1$. If some of the dimensions are empty, the energy allocated to nonempty dimensions will increase. This acts as a scale factor on the corresponding spectrum.

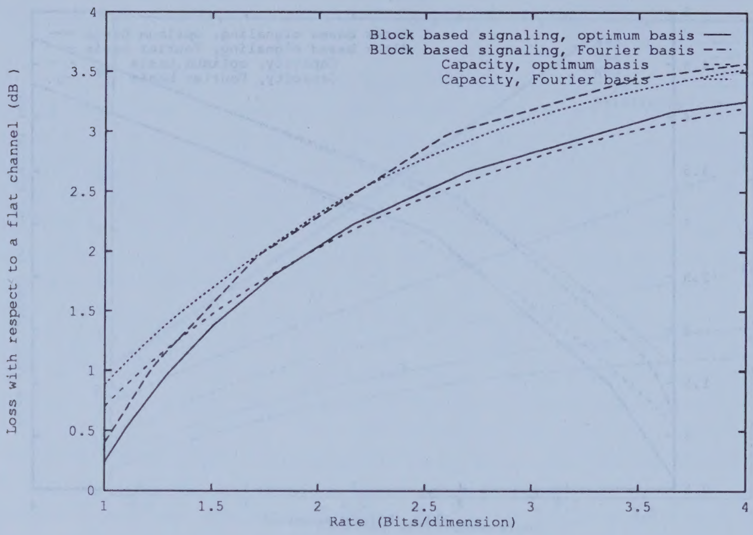


Fig. 4 Loss of energy in 1±D channel, $N = 24$.

6. Power spectrum

Let $\mathbf{x}$ be an M dimensional vector of time multiplexed impulses. The correlation matrix of $\mathbf{x}$ is defined as,

$$\mathbf{R}_x^k = \mathbf{E}\left[\mathbf{x}^0(\mathbf{x}^k)^*\right] , \tag{14}$$

where $\mathbf{x}^k$ is the vector at time k, $\mathbf{E}\,[\,.\,]$ is the expectation and * is the Hermitian transpose. The power spectrum of $\mathbf{x}$ is equal to, [9],

$$S_x(\omega) = \frac{1}{M}\sum_{k=-\infty}^{\infty} e^{-jk\omega M}\mathbf{V}^*(\omega)\mathbf{R}_x^k\mathbf{V}(\omega) , \qquad |\omega| \le \pi , \tag{15}$$

where,

$$\mathbf{V}(\omega) = \left[e^{-j\omega m},\ m = 0, \ldots, M-1\right]^t . \tag{16}$$

We have $\mathbf{R}_x^k = (\mathbf{R}_x^k)^*$. For a real $\mathbf{x}$, we have $\mathbf{R}_x^k = (\mathbf{R}_x^k)^t$.

In a block coding scheme, subsequent blocks are uncorrelated, i.e., $\mathbf{R}_x^k = 0,\ k \neq 0$. In this case, omitting the time index, (15) reduces to,

$$S_x(\omega) = \frac{1}{M}\sum_{m=0}^{M-1}\sum_{n=0}^{M-1} R_x(m,n)\cos\left[\omega(m-n)\right] , \tag{17}$$

where $R_x(m,n)$ is the (m,n)'th element of $\mathbf{R}_x$.

6..1 Baseband power spectrum

Modulation along each dimension can be considered as the multiplication of a baseband signal with the unit vector of the corresponding modulator dimension. Each baseband signal is obtained by passing the corresponding component of $\mathbf{a}$ through a sample and

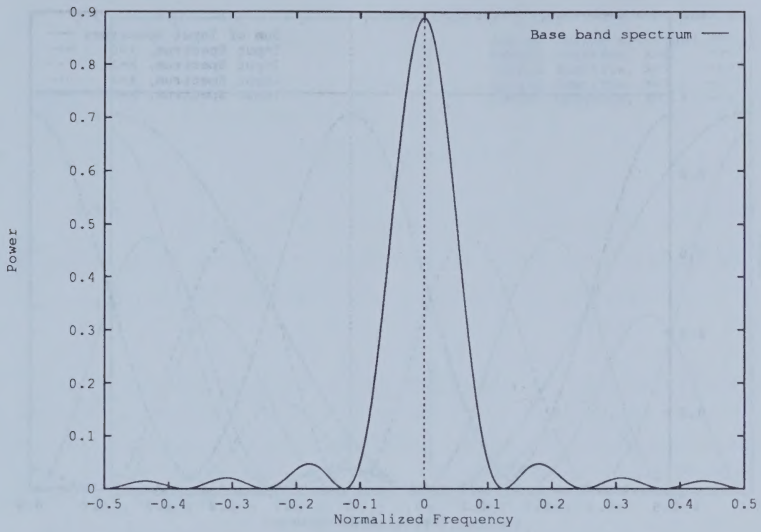


Fig. 5 Spectrum of the baseband signal.

hold system. The amplitude is hold for N periods and is followed by a single zero. For a total power of one, the spectrum of this signal is equal to,

$$S_B(\omega) = \frac{1}{N(N+1)} \frac{\sin^2(N\omega/2)}{\sin^2(\omega/2)} . \tag{18}$$

This spectrum is shown in Fig. 5.

We can obtain an alternative representation of (18) by considering that the baseband correlation matrix, $\mathbf{R}_B$, is an $(N+1) \times (N+1)$ matrix with the elements,

$$R_B(m,n) = \begin{cases} 1/N & 0 \le m,n \le N-1\,, \\ 0 & \text{otherwise}\,. \end{cases} \tag{19}$$

Substituting $M = N+1$ and (19) in (17), we obtain,

$$S_B(\omega) = \frac{1}{N(N+1)} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} \cos\left[\omega(m-n)\right] . \tag{20}$$

6..2 Power spectrum at the channel input

For the process $\mathbf{x}$, we have,

$$\mathbf{R}_x = \mathbf{M}\mathbf{R}_a\mathbf{M}^t . \tag{21}$$

To calculate the spectrum of the k'th dimension, we assign power $(N+1)/N$ to the k'th component of $\mathbf{a}$ and zero power to other components. In this case, using (2) and (21), the correlation matrix of the k'th dimension is found as,

$$R_{x_k}(m,n) = \frac{2}{N} \sin\frac{\pi(k+1)(m+1)}{N+1} \sin\frac{\pi(k+1)(n+1)}{N+1} , \tag{22}$$

where $0 \le m,n,k \le N-1$. This can be used in (17) to calculate the power spectrum. Figure 6 shows the spectrum of different dimensions of $\mathbf{x}$ for N=4. The sum of the spectrums for $\mathbf{x}$ is flat and equal to one.

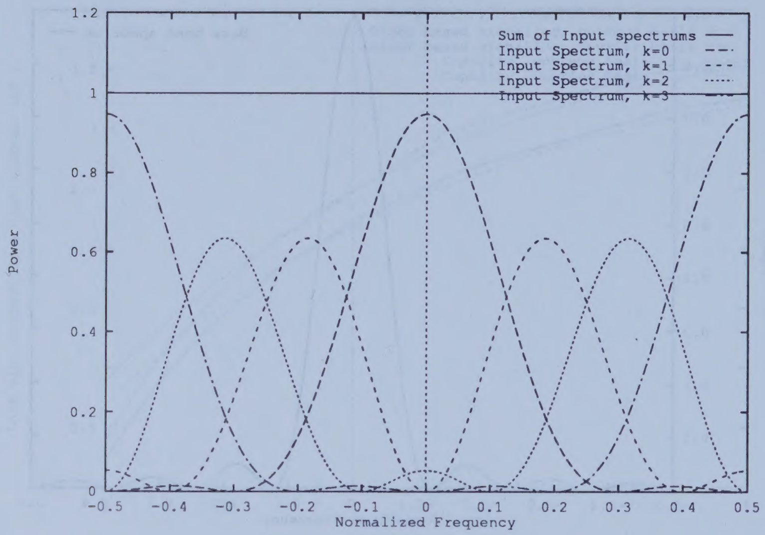


Fig. 6 Spectrum of the optimum input basis (input eigenvectors) for 1±D channels.

6..3 Power spectrum at the channel output

The basis at the channel output are the output eigenvectors of the channel. It can be shown that for the k'th dimension, the elements of $\mathbf{R}_{y_k}$ are equal to,

$$R_{y_k}(m,n) = \frac{1}{N}\sum_{i=0}^{1}\sum_{j=0}^{1}(-1)^{i+j}\,\sin\frac{\pi(k+1)(m+i)}{N+1}\,\sin\frac{\pi(k+1)(n+j)}{N+1}\,, \tag{23}$$

which can be simplified as,

$$R_{y_k}(m,n)=\frac{2}{N}\left[1-\cos\frac{\pi(k+1)}{N+1}\right]\cos\frac{\pi(k+1)(m+0.5)}{N+1}\,\cos\frac{\pi(k+1)(n+0.5)}{N+1}\,, \tag{24}$$

where $0 \le m,n \le N$ and $0 \le k \le N-1$. These expressions can be used in (17) to calculate the power spectrum. The result of these calculations for $N = 4$ is shown in Fig. 7. The sum of these spectrums is equal to $1-\cos(\omega)$ which is the power spectrum of the channel impulse response.

7. Summary

We have studied the optimum block based signaling over the partial response channels. In specific, channels 1±D and $1-D^2$ are studied. Analytical expressions are found for the eigenvalues/eigenvectors of these channels. The eigenvectors are either sine function or closely related to it. This reduces the computational complexity of the modulation by the use of the fast sine transform algorithms. Optimum basis is compared with the Fourier basis. Analytical expression is found for the capacity of these channels.

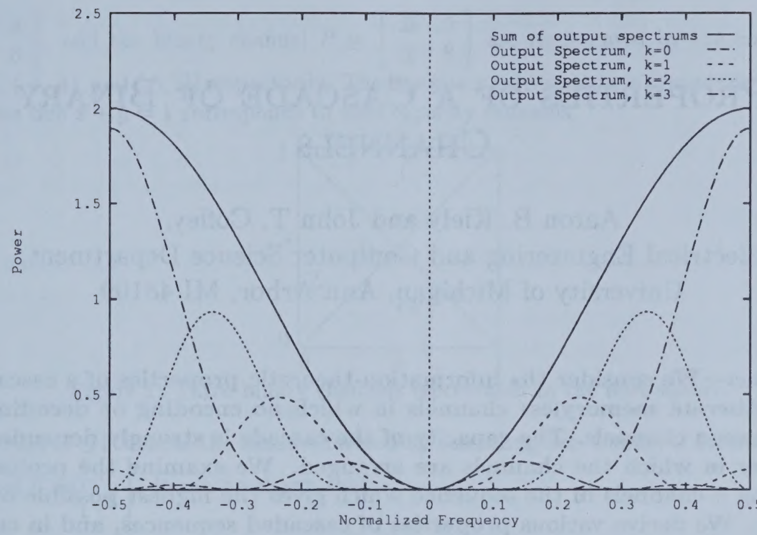


Fig. 7 Power spectrum at the output of the 1±D channels (optimum basis).

References

[1] G. D. Forney and A. R. Calderbank, "Coset codes for partial response channels; or, coset codes with spectral nulls," *IEEE Trans. Inform. Theory,* vol. IT–35, pp. 925–943, September 1989.

[2] A. K. Khandani, P. Kabal and H. Leib, "Combined coding and shaping over a multitone channel," To be presented at the *IEEE Global Telecommunications Conference, 1991*

[3] A. K. Khandani and P. Kabal "Shaping multi-dimensional signal constellations," *IEEE Int. Symp. Inform. Theory,* p. 4, June 1991, Budapest, Hungary.

[4] S. Kasturia, J. T. Aslanis, and J. M. Cioffi, "Vector coding for partial response channels," *IEEE Trans. Inform. Theory,* vol. IT–36, pp. 741–762, July 1989.

[5] P. Kabal and S. Pasupathy, "Partial response signaling," *IEEE Trans. Commun.,* vol. 23, pp. 921–934, September 1975.

[6] R. G. Gallager, *Information theory and reliable communication,* John Wiley & Sons, New York, NY, 1968.

[7] J. M. Wozencraft and I. M. Jacobs, *Principles of Communication Engineering,,* John Wiley & Sons, New York, NY, 1965.

[8] G. D. Forney, Jr., R. G. Gallager, G. R. Lang, F. M. Longstaff , and S. U. Quereshi, "Efficient modulation for bandlimited channels," *IEEE J. Select. Areas Commun.,* vol. SAC–2, pp. 632–647, September 1984.

[9] G. L. Cariolaro and G. P. Tronca, "Spectra of block coded digital signals," *IEEE Trans. Commun.,* vol. COM–22, pp. 1555–1563, Oct 1974.

PROPERTIES OF A CASCADE OF BINARY CHANNELS

Aaron B. Kiely and John T. Coffey,
Electrical Engineering and Computer Science Department,
University of Michigan, Ann Arbor, MI 48109

Abstract—**We consider the information-theoretic properties of a cascade of binary discrete memoryless channels in which no encoding or decoding occurs between channels. The capacity of the cascade is strongly dependent on the order in which the channels are arranged. We examine the problem of cascading n channels in the sequence which gives the highest possible overall capacity. We derive various properties of cascaded sequences, and in certain cases, these properties can be used to immediately specify the optimal sequence. Finally, these properties are used to obtain bounds on the capacity of a cascade of channels.**

1 Introduction

When communicating through a cascade of discrete memoryless channels (DMCs) without any re-encoding between successive channels, the capacity of the overall cascade is strongly dependent on the order in which the channels are placed. In this paper, we are interested in the following problem: given a set of DMCs $\{C_i\}_{i=1}^{n}$, in what order should we arrange these channels to achieve the highest overall capacity? We can use the convexity and symmetry of the capacity function to derive a number of properties which must be satisfied by the capacity optimal cascade, and in certain cases, we can use these properties to immediately specify the optimal cascade. For simplicity, we will focus exclusively on binary channels.

Such problems model the relaying of information from node-to-node or user-to-user in a communications network. We can also consider these questions as aspects of data processing, where each processing step is modeled as a DMC. The above questions can then be seen in the same context as the classic data processing inequality.

2 Basic Properties of Cascades

2.1 Graphical Representation

We represent a binary DMC $\begin{bmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{bmatrix}$ by the vector $\begin{bmatrix} \alpha \\ \beta \end{bmatrix}$, i.e., the point in the unit square whose coordinates are the transition error probabilities for the channel. For example, in Figure 1, the Z channel $Z = \begin{bmatrix} 1 & 0 \\ .3 & .7 \end{bmatrix}$, the symmetric channel $B =$

$\begin{bmatrix} .6 & .4 \\ .4 & .6 \end{bmatrix}$, and the binary channel $P = \begin{bmatrix} .5 & .5 \\ .2 & .8 \end{bmatrix}$ are represented by the coordinates $(0, .3), (.4, .4)$, and $(.5, .2)$ respectively. The line $y = x$ corresponds to symmetric channels, and the line $x + y = 1$ corresponds to zero capacity channels.

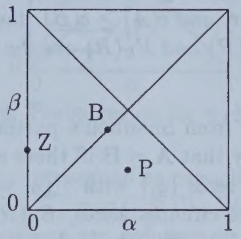


Figure 1: Three binary channels represented in the unit square.

We let $C \odot \Gamma$ denote the channel formed by cascading the channel C followed by the channel $\Gamma = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$. It can be shown that

$$C \odot \Gamma = \Gamma + m(\Gamma)C \tag{1}$$

or equivalently,

$$C \odot \Gamma = C + (1 - m(\Gamma))[Q(\Gamma) - C] \tag{2}$$

where $m(\Gamma) \triangleq 1 - \alpha - \beta$, which is the determinant of the channel matrix of Γ, and $Q(\Gamma) \triangleq (1/(\alpha + \beta))\Gamma$, which is the projection of Γ onto the zero capacity line. Figure 2 gives a graphical representation of $C \odot \Gamma$. A channel for which m is positive is called a "near side" channel (because it lies on the side of the zero capacity line closest to the origin), and a channel for which m is negative is called a "far side" channel.

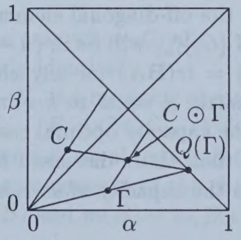


Figure 2: Graphical representation of $C \odot \Gamma$.

Consider the locations of the $n!$ points corresponding to each possible cascade that uses each channel in $\{C_i\}_{i=1}^{n}$ exactly once. The determinant of the overall channel matrix for such a cascaded channel will be independent of the order in which the channels are placed. Consequently, all of the possible cascaded channels will lie on a line in the unit square having slope -1. From the convexity and symmetry of the capacity function, the overall cascade having highest capacity will be the one that lies the farthest from the BSC line. Hence if we define $r(\Gamma) \triangleq \alpha - \beta$, then we need only consider the sequence which maximizes $|r|$ for the overall cascade. We refer to the sequence which maximizes (minimizes) r as the rmax (rmin) optimal sequence.

The following two results are consequences of the fact that all possible cascades lie on the same line having slope -1:

Lemma 1 *If A is a binary channel formed by cascading several channels, B is formed by cascading the same channels in a different order, and $c(A) = c(B)$, then A is identical to the channel B after a re-labeling of the inputs and outputs of B.*

Theorem 1 *If A is formed by cascading several channels, B is formed by cascading the same channels in a different order, and $c(A) \geq c(B)$, then $E_A(R) \geq E_B(R)$ for all rates R less than capacity. (Where $E_A(R)$ and $E_B(R)$ are the reliability functions of channels A and B respectively.)*

Proof: This theorem follows from Shannon's partial ordering of channels [3]. For two channel matrices $\mathbf{A}, \mathbf{B}$, we say that $\mathbf{A} \preceq \mathbf{B}$ iff there exists sets of stochastic matrices $\{\mathbf{R}_i\}, \{\mathbf{T}_i\}$, and a set of positive reals $\{g_i\}$ with $\sum_i g_i = 1$ such that $\mathbf{A} = \sum_i g_i \mathbf{R}_i \mathbf{B} \mathbf{T}_i$. Shannon showed that under these circumstances, $E_A(R) \geq E_B(R)$ for all rates R less than capacity. The "Shannon region" of a channel A (i.e., the set of all channels B for which $A \preceq B$ is determined as in Figure 3. It includes all channels on the same line of slope -1 as A that are closer than A to the BSC line, hence it includes all cascades having lower capacity. ∎

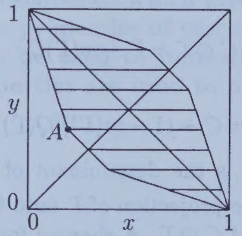


Figure 3: Shannon region for a channel A.

For $k \times k$ channels, if we represent a channel by the point in a $k(k-1)$ dimensional hypercube whose coordinates are the off-diagonal elements of the channel matrix, then in general the possible cascades of $\{C_i\}_{i=1}^n$ will lie in $(n-1)!$ different hyperplanes. This arises from the fact that $tr(\mathbf{AB}) = tr(\mathbf{BA})$ for any channel matrices $\mathbf{A}$ and $\mathbf{B}$, and because the trace of a channel matrix is equal to k minus the sum of the off-diagonal elements. Consequently, finding the capacity optimal cascade for sets of channels where $k > 2$ requires methods different than those discussed here, although these techniques can be used to produce bounds on the capacity of such a cascade.

2.2 Prefix Region

For a channel Γ, we define the *prefix region* $P(\Gamma)$ by

$$P(\Gamma) \triangleq \{C : c(C \odot \Gamma) \geq c(\Gamma \odot C)\} \tag{3}$$

where c is the capacity function. Thus $P(\Gamma)$ is the set of all channels C that should precede Γ to maximize the capacity of the cascaded pair Γ, C. The prefix region of a channel is shown in Figure 4. A necessary condition for $C_1, C_2, \ldots, C_n$ to be the capacity optimal sequence is that

$$C_1 \odot C_2 \odot \cdots C_j \in P(C_{j+1} \odot C_{j+2} \odot \cdots C_n) \tag{4}$$

for all $j = 1, 2, \ldots n-1$.

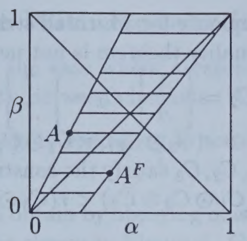


Figure 4: Prefix region for a channel A.

The edges of the prefix region correspond to capacity commutative channels, i.e., the set of channels C such that $c(\Gamma \odot C) = c(C \odot \Gamma)$. We find that Γ and C are capacity commutative iff their channel vectors are colinear or $r(C)/(1 + m(C)) = -r(\Gamma)/(1 + m(\Gamma))$. The prefix relation is nontransitive.

3 Properties of Optimal Cascades

We can exploit the symmetry and convexity of the capacity function to derive properties related to subsequences of an optimal cascade.

Theorem 2 (subsequence optimality) *If $C_1, C_2, \ldots, C_n$ is the rmax (rmin) optimal arrangement of $\{C_i\}_{i=1}^n$, then for any subsequence $C_i, C_{i+1} \ldots C_j$, $1 \leq i < j \leq n$,*

1. *if $C_{j+1} \odot C_{j+2} \odot \cdots C_n$ is a near side channel, or $j = n$, then $C_i, C_{i+1} \ldots C_j$ is the rmax (rmin) optimal arrangement of $\{C_k\}_{k=i}^j$,*
2. *if $C_j \odot C_{j+1} \odot \cdots C_n$ is a far side channel, then $C_i, C_{i+1} \ldots C_j$ is the rmin (rmax) optimal arrangement of $\{C_k\}_{k=i}^j$.*

Proof: We consider only for the case where $C_1, C_2, \ldots, C_n$ is rmax optimal. The proof for the rmin optimal case is nearly identical.

Suppose that $C_{j+1} \odot C_{j+2} \odot \cdots C_n$ is a near side channel but $C_i, C_{i+1}, \ldots C_j$ is not rmax optimal. Let $A = C_1 \odot C_2 \odot \cdots C_{i-1}$, $B = C_i \odot C_{i+1} \odot \cdots C_j$, and $C = C_{j+1} \odot C_{j+2} \odot \cdots C_n$ so that the overall cascaded channel is equal to $A \odot B \odot C$. Now if $C_i, C_{i+1}, \ldots C_j$ is not rmax optimal, then by some re-arrangement of the channels in $\{C_k\}_{k=i}^j$ we can form a channel B' such that $r(B') > r(B)$ and $m(B) = m(B')$. Then

$$r(A \odot B) = r(B) + m(B)r(A) < r(B') + m(B')r(A) = r(A \odot B'). \tag{5}$$

But $r(A \odot B) < r(A \odot B')$ implies

$$r(A \odot B \odot C) = r(C) + m(C)r(A \odot B) < r(C) + m(C)r(A \odot B') = r(A \odot B' \odot C) \tag{6}$$

since $m(C) > 0$. But $A \odot B' \odot C$ is a channel formed by re-arranging the channels of $\{C_i\}_{i=1}^n$, hence the original sequence is not be rmax optimal.

A nearly identical argument holds for the case where $C_{j+1} \odot C_{j+2} \odot \cdots C_n$ is a far side channel.

∎

While the subsequence optimality theorem provides necessary and sufficient conditions for a sequence to be rmax (rmin) optimal (since trivially we can take the entire sequence as a subsequence), a sequence for which all strict subsequences satisfy the constraints of the subsequence optimality theorem is not necessarily rmax (rmin) optimal.

For example, if $C_1 = \begin{bmatrix} .9 \\ .3 \end{bmatrix}$, $C_2 = \begin{bmatrix} .1 \\ .4 \end{bmatrix}$, $C_3 = \begin{bmatrix} .7 \\ .7 \end{bmatrix}$, then through direct calculation it can be verified that $r(C_2 \odot C_3) > r(C_3 \odot C_2)$, $r(C_1 \odot C_2) < r(C_2 \odot C_1)$, and $m(C_3) < 0$, thus all strict subsequences of C_1, C_2, C_3 satisfy the constraints of the (rmax) subsequence optimality theorem. However, $r(C_1 \odot C_2 \odot C_3) < r(C_3 \odot C_2 \odot C_1)$, thus C_1, C_2, C_3 is not rmax optimal.

For a channel Γ, we define $d(\Gamma) \triangleq r(\Gamma)/(1 - m(\Gamma))$, which is an increasing function of the angle between the channel and the BSC line, (where we take channels having positive r to have a positive angle). For the remainder of this paper, we assume that all channels under consideration have m strictly less than one, so that d is always well defined. The following corollary is useful in determining the rmax optimal sequence in certain special cases discussed in Section 4.

Corollary 3 *If $C_1, C_2, \ldots, C_n$ is rmax optimal, then for any two adjacent subsequences $C_i, C_{i+1}, \ldots C_j$ and $C_{j+1}, C_{j+2}, \ldots C_k$, $1 \le i \le j < k \le n$ we have*

$$d(C_i \odot C_{i+1} \odot \cdots C_j) < d(C_{j+1} \odot C_{j+2} \odot \cdots C_k) \tag{7}$$

iff $C_{k+1} \odot C_{k+2} \odot \cdots C_n$ is a near side channel or $k = n$. The converse holds for rmin optimal sequences. (We assume here that $d(C_i \odot C_{i+1} \odot \cdots C_j) \neq d(C_{j+1} \odot C_{j+2} \odot \cdots C_k)$).

This follows from subsequence optimality and the easily verified fact:

$$r(C \odot \Gamma) < r(\Gamma \odot C) \text{ iff } d(\Gamma) < d(C). \tag{8}$$

4 Optimal Sequences: Special Cases

Corollary 3 provides sufficient constraints to specify the optimal sequence in the following special cases.

4.1 Optimal Sequence for Near Side Channels

If we have a set of near side channels, we can specify the optimal ordering simply by knowing the angle of each channel.

Theorem 4 *If $\{C_i\}_{i=1}^n$ consists entirely of near side channels, then $C_1, C_2, \ldots, C_n$ is rmax optimal iff it is arranged in order of increasing angle. Conversely, $C_1, C_2, \ldots, C_n$ is rmin optimal iff it is arranged in order of decreasing angle. Hence for a set of near side channels, the capacity optimal order is to arrange the channels either by increasing or decreasing angle.*

Proof: Suppose $C_1, C_2, \ldots, C_n$ is rmax optimal but is not arranged in order of increasing angle. Then $d(C_i) > d(C_{i+1})$ for some $i < n$. But since each channel in $\{C_i\}_{i=1}^n$ has positive m, $m(C_{j+2} \odot C_{j+3} \odot \cdots C_n) = \prod_{k=j+1}^n m(C_k) > 0$, contradicting Corollary 3. ■

4.2 Optimal Sequence With Re-labeling

So far we have considered only the case where a cascade must be formed without any re-labeling of inputs and outputs between channels. A relabeling of the inputs and/or outputs of a channel is equivalent to cascading a *noiseless inverter* channel $R \triangleq \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ before and/or after the channel.

The possible channels we can obtain by applying noiseless inverters to a channel A are denoted by A^F, A^R, and A^{RF}, as shown in Figure 5. Graphically, the effect of applying noiseless inverters to a channel is to reflect the channel about the zero capacity line, the BSC line, or both.

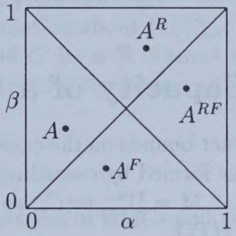


Figure 5: Applying noiseless inverters to a channel A.

If we are to communicate through a cascade of $\{C_i\}_{i=1}^n$ and arbitrarily many inverters, we want to know how to arrange the channels and inverters to maximize the capacity of the overall cascade. We can think of the overall capacity optimal cascade as a cascade of the channels $\{C_i'\}_{i=1}^n$, where each C_i' is one of $\{C_i, C_i^F, C_i^R, C_i^{RF}\}$. Thus, we want to know how inverters should be applied to each channel individually, and in what order to place the modified channels.

Theorem 5 *If we are to use inverters to maximize the capacity of the cascade of $\{C_i\}_{i=1}^n$, the highest possible overall capacity can be obtained by arranging the channels as follows:*

1. *Apply inverters as necessary to form the set of modified channels $\{C_i'\}_{i=1}^n$ such that each modified channel is a near side channel, and has positive r (i.e., the modified channel lies below the BSC line).*
2. *Arrange the modified channels in order of increasing angle.*

This theorem follows from Corollary 3 and induction on the following lemma:

Lemma 2 *If Γ and C are near side channels such that $0 < d(\Gamma) < d(C)$, then $\Gamma \odot C$ is the channel having highest possible capacity of all channels obtained by cascading Γ, C and arbitrarily many noiseless inverters.*

Proof: Since applying inverters to the beginning or end of a cascade does not affect the overall capacity, we need only consider the cascaded channels $\Gamma \odot C$, $C \odot \Gamma$, $\Gamma \odot R \odot C$, and $C \odot R \odot \Gamma$. We note that $m(\Gamma \odot C) = m(C \odot \Gamma) = -m(\Gamma \odot R \odot C) = -m(C \odot R \odot \Gamma)$, hence from the symmetry and convexity of the capacity function, the channel having the highest capacity will be the one having the largest value of $|r|$. Evaluating r for each of these channels gives

$$r(\Gamma \odot C) = r(C) + m(C) r(\Gamma) \tag{9}$$

$$r(C \odot \Gamma) = r(\Gamma) + m(\Gamma) r(C) \tag{10}$$

$$r(\Gamma \odot R \odot C) = r(C) - m(C) r(\Gamma) \tag{11}$$

$$r(C \odot R \odot \Gamma) = r(\Gamma) - m(\Gamma) r(C). \tag{12}$$

Now $r(C)$, $r(\Gamma)$, $m(C)$, and $m(\Gamma)$ are all positive, so by (11) and (12), neither $C \odot R \odot \Gamma$ nor $\Gamma \odot R \odot C$ have the largest value of $|r|$. Further, from (8), $r(\Gamma \odot C) > r(C \odot \Gamma)$, and since both of these quantities are positive we conclude that $\Gamma \odot C$ has the highest capacity. ■

Any capacity optimal cascade can be represented as a cascade of modified channels. However, neither the optimal cascade nor its representation as a cascade of modified channels is unique. The arrangement described in Theorem 5 does not necessarily minimize the number of inverters used.

5 Bounds on the Capacity of a Cascade

We now use these results to construct bounds on the capacity of a cascade of $\{C_i\}_{i=1}^n$. Let $\{P_i\}_{i=1}^{n!}$ be the $n!$ possible channels formed by cascading each channel in $\{C_i\}_{i=1}^n$exactly once (without any re-labeling). Let $M = \prod_{i=1}^n m(C_i)$ so $m(P_i) = M$ for all $i = 1, \ldots n!$. We have the following bounds on $c(P_k)$:

1. From the well known data processing inequality,

$$c(P_k) \le \min_i \{c(Ci)\} \tag{13}$$

   with equality iff at least $n-1$ of the channels are noiseless.

2. Since for any fixed line of constant $\alpha + \beta$ (other than the zero capacity line) the channels having the lowest and highest capacity are the BSC and Z channel respectively, we have

$$1 - \mathcal{H}_2(T/2) \le c(P_k) \le \log_2 \left(1 + \exp_2 \left[\frac{\mathcal{H}_2(T)}{T-1} \right] \right) \tag{14}$$

   where $T \triangleq 1 - M$, and $\mathcal{H}_2(x) \triangleq -x \log_2(x) - (1-x)\log_2(1-x)$, which is the binary entropy function. The lower bound is achieved iff P_i is a BSC, and the upper bound is achieved iff P_i is a Z channel (i.e., all of the channels in $\{C_i\}_{i=1}^n$ are Z channels).

3. If $\{C_i\}_{i=1}^n$ consists only of near side channels, then

$$\min_j \{d(C_j)\} \le d(P_k) \le \max_j \{d(C_j)\}, \tag{15}$$

   which follows by induction on the fact that

$$d(\Gamma) < d(C \odot \Gamma) < d(\Gamma \odot C) < d(C) \tag{16}$$

   for any two near side channels C, Γ such that $d(\Gamma) < d(C)$. Thus $c(P_k)$ cannot exceed the capacity of a channel X satisfying $m(X) = M$ and $d(X) = \max_j\{|d(C_j)|\}$, i.e.,

$$c(P_k) \le c\left[\left(\frac{1-M}{1-m(Cr)} \right) C_r \right] = c[X] \tag{17}$$

   where C_r is the channel in $\{C_i\}_{i=1}^n$having the largest value of $|d|$.

4. Alternatively, we can use Theorem 5 to derive a similar bound which holds for any set of channels $\{C_i\}_{i=1}^n$. Note that the modified channels $\{C_i'\}_{i=1}^n$ are all near side channels, hence we can apply the bound (17) to obtain

$$c(P_k) \leq c\left[\left(\frac{1-M}{1-m(Cz)}\right) C_z\right] \tag{18}$$

where $d(C_z) = \max_i d(C_i')$. Since the optimal ordering with inverters has capacity at least as high as that without inverters, this bound applies to cascades with or without inverters.

In general the data processing inequality gives the least tight bound because it makes use of the least amount of information about $\{C_i\}_{i=1}^n$. For example, consider the bounds (13) and (17). Let the channel G be a Z channel such that $c(G) = \min_i[c(C_i)]$. If $|M| < |m(G)|$ (which we would expect for large n), then (17) is tighter than (13).

6 Conclusion

A convenient graphical representation of DMCs and their cascades along with symmetry and convexity properties of the capacity function can be used to derive various properties related to the capacity of a cascade of channels. These properties can be used to immediately specify the capacity optimal cascade of a set of channels in certain special cases, and can be used to derive bounds on the capacity of a cascade of channels.

References

[1] A. B. Kiely, J. T. Coffey, and M. R. Bell, "On the Capacity of a Cascade of Channels," *Proc. 1991 IEEE Intl. Symp. Inform. Theory*, p. 60.

[2] S. Muroga, "On the Capacity of a Discrete Channel. I." *J. Phys. Soc. Japan* vol. 8, pp. 484-494, 1953.

[3] C. E. Shannon, "A Note on Partial Ordering for Communication Channels," *Inform. Contr.*, vol. 1, pp. 390-398, 1958.

[4] R. A. Silverman, "On Binary Channels and their Cascades," *IRE Trans. Inform. Theory*, vol. 1, pp. 19-27, 1955.

Unsymmetrical Boundary Shaping in Multidimensional Spaces

A. K. Khandani[1], P. Kabal[1,2]

[1]Dept. of Elec. Eng. McGill University, 3480 University St., Montreal, Canada, H3A-2A7
[2]INRS-Telecommunications, 3 Place du Commerce, Verdun, Canada, H3E-1H6

Abstract: The power spectrum of a signal constellation depends on the constellation dimensions (modulating waveforms) and to the power allocated to each of them. For a given orthonormal basis, spectral shaping is achieved by allocating different powers to different dimensions. In this work, the idea of unsymmetrical shaping is introduced. This is the selection of the boundary of a signal constellation which has different values of power along different dimensions. For a given set of powers, λ_i's, shaping is achieved by selecting a region which has the second moment λ_i along the i'th dimension and its volume is as large as possible. This is equivalent to maximizing the shape gain. A larger shape gain is obtained at the price of a larger constellation expansion ratio, CER$_s$. The structure of the region which optimizes the tradeoff between CER$_s$ and γ_s is discussed. A practical addressing scheme is given to achieve a point near the knee of the corresponding tradeoff curves.

1. Introduction

In a one dimensional constellation, the outer points are of higher energy. By using these points less frequently, one can decrease the average energy of the set for a given entropy, [1]. Selecting an appropriate boundary in a space of higher dimensionality and using the points inside of that boundary with equal probability is a practical way of inducing a nonuniform probability distribution on the space dimensions. This is a practical method because the higher dimensional points are used with equal probability. This is the basic idea of shaping. Another practical method for inducing a nonuniform probability distribution on the space dimensions is given in [1].

On the other hand, the joint probability distribution and consequently the correlation between the dimensions also depends on the constellation boundary. This can be used to shape the spectrum of the resulting process by the appropriate selection of the shaping region. In general, a nonflat spectrum is associated with an unsymmetrical region. This is a region which has different values of second moment along different dimensions. The second moment along the i'th dimension is denoted by λ_i. For a given $\sum_i \lambda_i$ (fixed total power), a symmetrical region corresponding to equal λ_i's has larger volume than an unsymmetrical region. The reduction in the volume (rate) is the price associated with a nonflat spectrum.

Figure 1 shows the block diagram of the system under consideration.

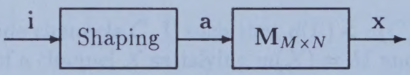


Fig. 1 System block diagram.

In each signaling interval which is composed of M time slots, a binary vector **i** is transmitted. This is the set of data bits and (possibly) some channel coding redundant bits. The shaping block maps the vector **i** to the point **a** in the baseband constellation. This is a set of N-dimensional points bounded within the shaping region $\mathcal{R}_a$. The $M \times N$ orthonormal matrix **M** maps the point **a** to the point **x** in the modulated constellation. This is a set of points in an N-dimensional subspace of the original M-dimensional space (corresponding to M time slots). The dimensions of the modulated constellation are along the columns of **M**. The shaping region of **x** is denoted by $\mathcal{R}_x$.

As we will see later, under a set of mild conditions, the spectrum of **a** is white. In this case, within a multiplicative factor, the spectrum of **x** depends only on the space dimensions and on the power allocated to each dimension. This means that the general shape of the spectrum is independent of the exact structure of $\mathcal{R}_a$. The structure of $\mathcal{R}_a$ determines a multiplicative factor for the spectrum. A nonflat spectrum in obtained by selecting **M** as a nondiagonal matrix and allocating different values of power to its dimensions.

It is assumed that the points of the constellation are used with equal probability. In continuous approximation, the distribution of the points is approximated by a continuous uniform density within the shaping region. This is a valid approximation for relatively high rate per each dimension. In this case, the rate and the energy per signaling interval are determined by the volume and second moment of the shaping region.

For a given set of powers, λ_i's, shaping is achieved by selecting $\mathcal{R}_a$ as a region which has the second moment λ_i along the i'th dimension and its volume is as large as possible. We refer to this as unsymmetrical shaping because in general the second moments along different dimensions are different. Unsymmetrical shaping can be also considered as a practical way of inducing a nonuniform marginal distribution of variance λ_i along the i'th dimension. Nonuniformity of the distributions is such that the points of higher energy are used less frequently.

For a given set of λ_i's, the shape gain (γ_s) of $\mathcal{R}_a$ is defined as the reduction in the average power comparing to using a hypercube with the same volume and with the second moments proportional to λ_i's as the shaping region. Using continuous approximation over the hypercube, we obtain,

$$\gamma_s(\mathcal{R}_a) = \frac{1}{12}\left[\frac{V^2(\mathcal{R}_a)}{\prod_{i=0}^{N-1}\lambda_i}\right]^{1/N}, \qquad (1)$$

where $V(\mathcal{R}_a)$ is the volume of $\mathcal{R}_a$. The Constellation-Expansion-Ratio (CER_s) is defined as the ratio of the average number of points per dimension to the minimum necessary number of points per dimension. If the projection of $\mathcal{R}_a$ on the i'th dimension is the region $[-L_i, L_i]$, CER_s is equal to,

$$\mathrm{CER}_s(\mathcal{R}_a) = 2\left[\frac{\prod_{i=0}^{N-1} L_i}{V(\mathcal{R}_a)}\right]^{1/N}. \qquad (2)$$

In the conventional cases, CER_s is measured on a two dimensional basis, [2]. This is restricted in application to even values of dimensionality. Although in QAM schemes the dimensionality is always even, in our modulating scheme this may not be the case.

A $\gamma_s > 1$ is achieved at the price of a $\mathrm{CER}_s > 1$ and there exist a tradeoff between these two factors. An optimally shaped region is a region which optimizes this tradeoff.

2. Unsymmetrical shaping

From (1) and (2), it is seen that γ_s and CER_s are invariant to the scaling of the coordinates. This means that from the shaping point of view, regions obtained by scaling have the same performance.

Another price to be paid for shaping is an increase in the factor Peak-to-Average-power-Ratio (PAR). If we define the PAR as the average of the peak energies per different dimensions to the average energy per dimension, it will be also invariant to scaling of the coordinates. But, if we define the PAR as the the ratio of the maximum of all the peak energies per different dimensions to the average energy per dimension, it will not be invariant to scaling.

Scaling invariance property allows us to relax the constraints on the individual λ_i's to a single constraint on $\sum_i \lambda_i$. This changes the problem of the unsymmetrical shaping to a conventional shaping problem. In this case, one tries to maximize the volume of the shaping region for a given total second moment. We assume that the region $\mathcal{R}_a$ is obtained by scaling of a region $\mathcal{B}$. For $\mathcal{B}$, we have $L_i = 1, \forall i$. This is denoted as a baseline region. Region $\mathcal{B}$ is a subset of the hypercube $[-1,1]^N$, N fold cartesian product of $[-1,1]$. We refer to this hypercube as the available signal space. An optimally shaped $\mathcal{B}$ should be selected as a subset of the available signal space which has the maximum volume for a given second moment and given CER_s. To select such region, we subdivide the hypercube by a set of concentric hyperspheres into the energy shells. As all the points in a shell have the same energy, a shell should be either completely inside or completely outside of the optimally shaped region. A close look reveals that the resulting region is symmetrical.

In general, for a symmetrical baseline region, substituting $L_i = 1, \forall i$ and $\lambda_i = E, \forall i$ in (1) and (2), we obtain,

$$\gamma_s(\mathcal{B}) = \frac{[V(\mathcal{B})]^{2/N}}{12E}, \tag{3}$$

$$\mathrm{CER}_s(\mathcal{B}) = 2\,[V(\mathcal{B})]^{-1/N}. \tag{4}$$

Combining (3) and (4) results in,

$$E = \frac{1}{3\gamma_s(\mathrm{CER}_s)^2}. \tag{5}$$

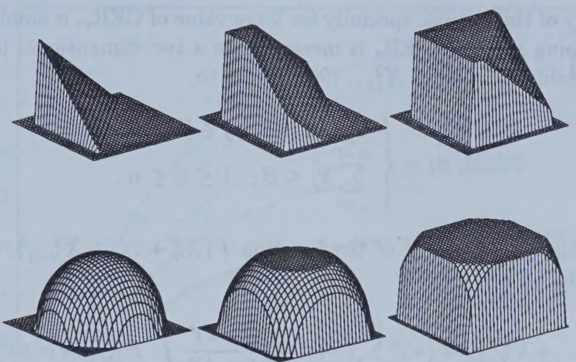
To have the second moment λ_i along the i'th dimension, the scale factors should be equal to,

$$S_i = \sqrt{\lambda_i/E} = \sqrt{3\gamma_s\lambda_i} \times \mathrm{CER}_s. \tag{6}$$

3. Two types of the baseline regions

3..1 Type I: optimally shaped region

For a baseline region, available signal space is a hypercube of edge length two centered at origin. Shaping is achieved by selecting a subset of this hypercube which has the largest volume for a given second moment and given CER_s. This subset is selected by a hypersphere. By changing the radius of the hypersphere, one can provide the desired tradeoff between γ_s and CER_s. The resulting region is denoted by $\mathcal{A}_N^1$.



$\beta = 1.0$ $\beta = 1.4$ $\beta = 2.1$

Fig. 2 $\mathbf{A}_N^1$ and $\mathbf{TC}_N$ regions for $N = 3$ and different values of β.

The dimensions of the space are labeled by X_i, $i = 0, \ldots, N-1$. Using this notation, we have,

$$\mathcal{A}_N^1 = \begin{cases} -1 \leq X_i \leq 1\,, & i \in [0, N-1]\,, \\ \sum\limits_{i=0}^{N-1} X_i^2 \leq \beta\,, & 1 \leq \beta \leq N\,, \end{cases} \tag{7}$$

where β is the radius of the hypersphere. In general, the projection of the region $\mathcal{A}_N^1$ on any N' dimensional subspace is equal to an $\mathcal{A}_{N'}^1$ region with the same value of β.

Using the change of variable $Y_i = X_i^2$, (7) reduces to,

$$\mathcal{TC}_N(1, \beta) = \begin{cases} Y_i \leq 1\,, & i \in [0, N-1]\,, \\ \sum\limits_{i=0}^{N-1} Y_i \leq \beta\,, & 1 \leq \beta \leq N\,. \end{cases} \tag{8}$$

This is a simplex of edge length β truncated within a hypercube of edge length one.

For $\beta \leq 1$, the $\mathcal{A}_N^1$ region is a sphere. This has the maximum γ_s (for a given N) but also large value for CER$_s$. For $\beta \geq N$, the $\mathcal{A}_N^1$ region is a hypercube. This corresponds to no shaping, i.e., γ_s=1 and CER$_s$=1. By changing β in the range $1 \leq \beta \leq N$, one can move along the optimum tradeoff curve between these two extreme points. Figure 2 shows the region $\mathcal{A}_3^1$ and the corresponding $\mathcal{TC}_3$ for three different values of β.

3..2 Type II: a suboptimum region

In the second type of baseline region, which is only for even values of N, a circle is used as the boundary of the two dimensional subspaces spanned by (X_0, X_1),. . .,(X_{N-2}, X_{N-1}). These are called the two dimensional subspaces and are indexed by $j = 0, \ldots, n-1, n = N/2$. The final region, which is denoted as $\mathcal{A}_N^2$, is located at the intersection of a hypersphere with the cartesian product of the circles, i.e.,

$$\mathcal{A}_N^2 = \begin{cases} X_j^2 + X_{j+1}^2 \leq 1\,, & j \in [0, n-1]\,, \\ \sum\limits_{i=0}^{N-1} X_i^2 \leq \beta\,, & 1 \leq \beta \leq n\,. \end{cases} \tag{9}$$

The suboptimality of this region, specially for large value of CER$_s$, is small. Indeed $\mathcal{A}_N^2$ is the optimum shaping region if CER$_s$ is measured on a two dimensional basis, [3]. Using the change of variable $Y_j = X_j^2 + X_{j+1}^2$, (9) reduces to,

$$\mathcal{TC}_n(1,\beta) = \begin{cases} Y_j \le 1\,, & j \in [0, n-1]\,, \\ \sum_{j=0}^{n-1} Y_j \le \beta\,, & 1 \le \beta \le n\,. \end{cases} \tag{10}$$

It can be shown that the integral of the function $F(X_0^2 + \ldots + X_{N-1}^2)$, over the region $\mathcal{A}_N^2$ is equal to, [3],

$$\int_{\mathcal{A}_N^2} F(X_0^2 + \ldots + X_{N-1}^2) = (\pi)^n \sum_{k=0}^{\lfloor \beta \rfloor} (-1)^k C_N^k \frac{(\beta - k)^n}{(n-1)!} \int_0^1 F\left[(\beta - k)\tau + k\right] \tau^n \, d\tau\,. \tag{11}$$

Calculation of (11) is based on expressing the region $\mathcal{TC}_n$ in (10) as the union and intersection of the simplexes and applying the Dirichelet's integral to each simplex.

A uniform density of points within the region $\mathcal{A}_N^2$ in (9) results in a uniform density of points within the region $\mathcal{TC}_n$ in (10). This is due to the fact that in a circle, the average energy and the volume are proportional. In this case, equal volume partitions of $\mathcal{TC}_n$ correspond to equal volume partitions of $\mathcal{A}_N^2$. This allows us to achieve the shaping/addressing on equal volume partitions of $\mathcal{TC}_n$. On the other hand, for $\beta = n/2$, region $\mathcal{TC}_n(1,\beta)$ is equal to the Voronoi region of the lattice D_n^* truncated to the positive coordinates. This allows us to use the Voronoi constellation based on the partition $Z^n + (1/2)^n / D_n^*$ for the addressing, [4]. This method achieves a point on the corresponding tradeoff curve which is near to the optimum curve.

3..3 Tradeoff between CER$_s$ and γ_s

An increase in γ_s is associated with an increase in CER$_s$. This means that there exists a tradeoff between these two factors. Equation (11) is used to calculate this tradeoff for the $\mathcal{A}_N^2$ region. Setting $F(X_0 + \ldots + X_{N-1}) = 1$ results in the volume and setting $F(X_0 + \ldots + X_{N-1}) = X_0^2 + \ldots + X_{N-1}^2$ results in the second moment. These are used in (3) and (4) to calculate CER$_s$ and γ_s. Figures 3 show the result of these calculations for $N = 8, 12, 16, 24, 32$. We will refer to the marked point on the curves later.

4. Addressing

Addressing is the assignment of the data bits to the constellation points. In a cubic constellation, addressing is achieved independently along each dimension. However, in a shaped constellation, independent addressing is not possible and a more complex, usually N-dimensional, addressing scheme is needed. The increase in the addressing complexity is a price to be paid for shaping.

4..1 Addressing in continuous approximation

We assume that the dimensionality is even, $N = 2n$. The shaping/addressing is achieved in two levels. The first level is on the two dimensional subspaces and the second level is on their cartesian product. By achieving the first level of addressing on a one dimensional basis, the same approach can be applied to an $\mathcal{A}_N^1$ region with odd dimensionality. First,

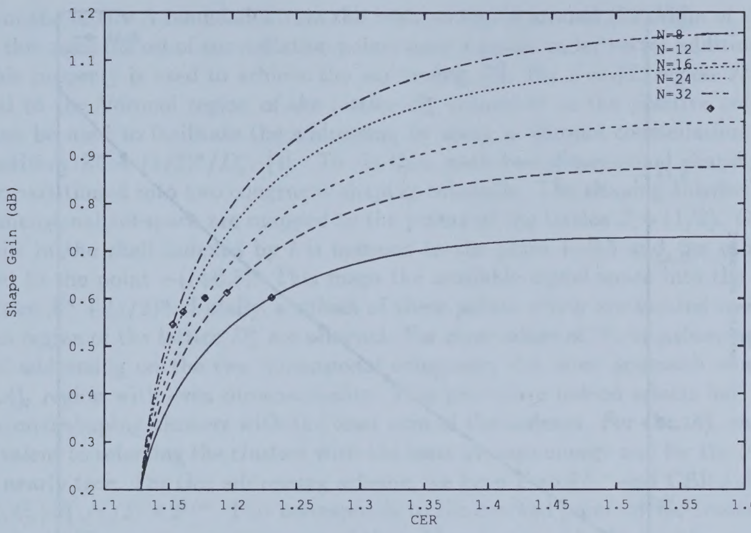


Fig. 3 Tradeoff curves for $\mathbf{A}_N^2$ regions.

by a set of concentric circles, each two dimensional subspace is divided into K subregions of equal volume. We refer to this subregions as the two dimensional shaping shells. These are indexed by $l=0,\ldots,K-1$. For the $\mathcal{A}_N^2$ region, the outer radius R, and the average energy E, of the l'th shell are equal to,

$$R_l = \sqrt{\frac{l+1}{K}} \quad \text{and} \quad E_l = \frac{l+1}{2K}\,. \tag{12}$$

From (12), it is seen that for the $\mathcal{A}_N^2$ region, the relationship between E_l and l is linear. Figure 4 shows the E_l as a function of l in the $\mathcal{A}_N^1$ region ($K = 64$). It is seen that in this case also this relationship is nearly linear.

The two dimensional shaping shells divide the available signal space into K^n, N-dimensional shaping clusters of equal volume. Each of these clusters is indexed by an index vector $\mathbf{l} = (l_j, j=0,\ldots,n-1)$ where l_j is the index of the two dimensional shaping shell in the j'th subspace. The second level of the shaping/addressing is achieved by selecting T of the N dimensional shaping clusters with the least average energy. For the $\mathcal{A}_N^2$ regions, because of the linear relationship between E_l and l in (12), this is achieved by selecting the clusters with the least sum of indexes. In the $\mathcal{A}_N^1$ regions, selecting the clusters with the minimum sum of indexes results in a small suboptimality. The CER$_s$ of this scheme is equal to,

$$\text{CER}_s(\mathcal{A}_N^1) = \frac{\sqrt{K}}{T^{1/N}}, \quad \text{and} \quad \text{CER}_s(\mathcal{A}_N^2) = \frac{\sqrt{\pi}}{2} \times \frac{\sqrt{K}}{T^{1/N}}\,. \tag{13}$$

4..2 Addressing in the discrete analysis

We assume that the coding lattice is $Z^N + (1/2)^N$ (no channel coding). The same discussion is valid if the coding lattice is a sublattice of $Z^N + (1/2)^N$. The main difference between the continuous approximation and the discrete case arises when we subdivide

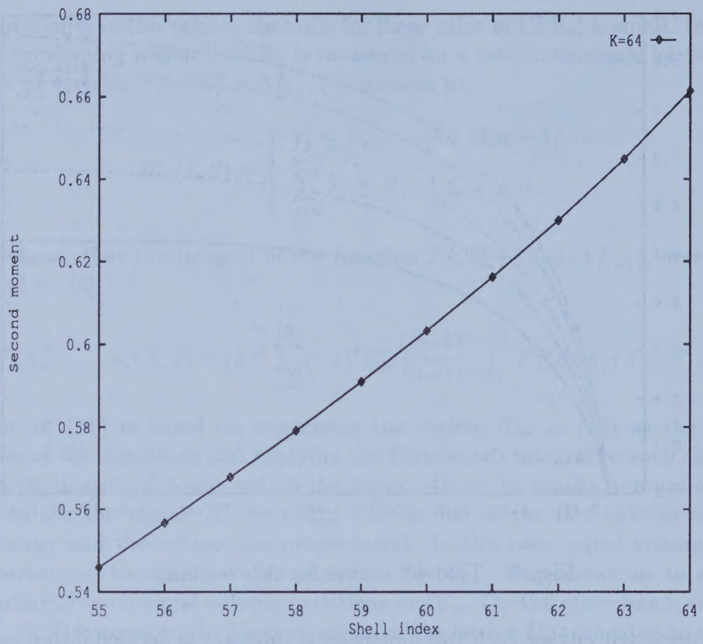


Fig. 4 Relation between E_l and l in $\mathbf{A}_N^1$ regions for $K = 64$.

the two dimensional subspaces into the shaping shells. In the discrete case, an equal number of points from $Z^2 + (1/2)^2$ in the order of the increasing energy are grouped together. There are P_j points in the shells of the j'th subspace. To transmit an integral number of bits per signaling interval, K and P_j should be an integral power of two, i.e., $K = 2^k$ and $P_j = 2^{p_j}$. This results in $K^n = 2^{kn}$, N-dimensional shaping clusters each containing $\prod_j P_j$ points (corresponding to the integral rate $\sum_j p_j$).

Shaping is achieved by selecting $T{=}2^t$ of the N-dimensional shaping clusters with the least average energy. Addressing can be achieved by a lookup table with t input bits and kn output bits. In each signaling interval, t bits enter the lookup table. The corresponding kn bits at the output are used to select a shaping shell within each two dimensional subspace. Another $\sum_j p_j$ data bits select a single point within each two dimensional shell selected by the previous part. In general, the tradeoff depends on the value of the P_j's. Decreasing the P_j's improves the tradeoff. For $P_j{=}4$, all the points within a two dimensional shell have the same energy. This results in the best possible tradeoff. In other words $P_j < 4$ achieves the same performance as $P_j = 4$.

In general, the set of approximations involved in the discrete case makes the difference between the two types of the shaping regions even less important. As an example of this effect, in dimensionality two and for some cardinalities, a circular and a rectangular shaping region result in the same constellations.

4..3 Shell-addressed Voronoi constellations

A Voronoi constellation based on the lattice partition Λ/Λ_s uses the set of the coset leaders of the corresponding partition as the constellation points. This is a subset of the points of the lattice Λ bounded within the Voronoi region around the origin of the lattice Λ_s. In this case, the set of constellation points form a group under vector addition modulo Λ_s. This property is used to achieve the addressing, [5]. For $\beta=n/2$, region $\mathcal{TC}_n$ in (10) is equal to the Voronoi region of the lattice D_n^* truncated to the positive coordinates. This can be used to facilitate the addressing by using a Voronoi constellation based on the partition $Z^n + (1/2)^n / D_n^*$, [4]. To do this, each two dimensional shaping shell is further partitioned into two congruent shaping subshells. The shaping subshells of each two dimensional subspace are mapped to the points of the lattice $Z + (1/2)$. One of the subshells in the shell indexed by l is mapped to the point l+0.5 and the other one is mapped to the point $-(l+0.5)$. This maps the available signal space into the points of the lattice $Z^n + (1/2)^n$. Finally, a subset of these points which are located inside of the Voronoi region of the lattice D_n^* are selected. For even values of N, by achieving the first level of addressing on the two dimensional subspaces, the same approach is applicable to an $\mathcal{A}_N^1$ region with even dimensionality. This procedure indeed selects half of the N dimensional shaping clusters with the least sum of the indexes. For the $\mathcal{A}_N^2$ region, this is equivalent to selecting the clusters with the least average energy and for the $\mathcal{A}_N^1$ region this is nearly true. For this addressing scheme, we have $T=0.5K^n$ and $\mathrm{CER}_s(\mathcal{A}_N^1)=2^{1/n}$, $\mathrm{CER}_s(\mathcal{A}_N^2)=(\sqrt{\pi}/2) \times 2^{1/n}$. This corresponds to the marked point on the tradeoff curves in Fig. 3. Unlike to the previous case of the addressing by a lookup table, generalizing this scheme to $\mathcal{A}_N^1$ regions with odd dimensionality results in some degradation.

5. Correlation matrix

Define the correlation matrix associated with the process $\mathbf{x}$ in Fig. 1 as,

$$\mathbf{R}_x = \mathbf{E}\left[\mathbf{x}\mathbf{x}^t\right] = \mathbf{M}\mathbf{R}_a\mathbf{M}^t \,, \tag{14}$$

where $\mathbf{E}[\ .\ ]$ denotes the expectation and $\mathbf{R}_a = \mathbf{E}\left[\mathbf{a}\mathbf{a}^t\right]$ is the correlation matrix of $\mathbf{a}$. The matrix $\mathbf{R}_a$ is determined by the structure of the region $\mathcal{R}_a$. For any shaping region which is not a hypercube, the probability distribution along different dimensions depend on each other. In spite of the dependency, under a set of mild conditions, the distributions are uncorrelated ($\mathbf{R}_a$ is diagonal).

We assume that subsequent $\mathbf{a}$'s and consequently subsequent $\mathbf{x}$'s are uncorrelated. In this case, the power spectrum of $\mathbf{x}$ is equal to, [6],

$$S_x(\omega) = \frac{1}{N}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} R_x(i,j)\cos\left[\omega(i-j)\right], \tag{15}$$

where $R_x(i,j)$ is the (i,j)'th element of $\mathbf{R}_x$. A diagonal $\mathbf{R}_x$ corresponds to a white spectrum. For an orthonormal $\mathbf{M}$, the $\mathbf{R}_x$ will be diagonal if: (i) $\mathbf{R}_a$ and $\mathbf{M}$ are both diagonal, or if (ii) $\mathbf{R}_a$ is diagonal with equal diagonal elements and $\mathbf{M}$ is a rectangular matrix, i.e., $M=N$. If the channel dimensions are used directly as the constellation dimensions, $\mathbf{M}$ will be diagonal. It is also easy to prove that: (i) if the region $\mathcal{R}_a$ is closed under the sign changings of the coordinates, $\mathbf{R}_a$ is diagonal, and (ii) if the region $\mathcal{R}_a$ is closed under permutations of coordinates, the diagonal elements of $\mathbf{R}_a$ are equal. For both types of the baseline regions under consideration, the correlation matrix is diagonal with the elements E given in (5). In this case, $\mathbf{R}_a$ is diagonal with the i'th diagonal element equal to λ_i. For a diagonal $\mathbf{R}_a$, the spectrum depends only on the space dimensions and to the power allocated to each dimension. This means that the shape of the spectrum does not depend on the exact structure of the baseline region. But, the volume of the signal space (rate) depends on the structure of the baseline region and is proportional to $(\gamma_s)^{N/2}$.

The shaping region for **x**, $\mathcal{R}_x$, is the image of the baseline region, $\mathcal{B}$, under **MS** where **S** is a diagonal matrix with the diagonal elements equal to the scale factors. By changing **M** and **S**, one can shape the spectrum of **x**. This can be interpreted in terms of changing the region $\mathcal{R}_x$. In the following we give an example.

5..1 Example

The output eigenvectors of $1-D$ channel form an orthonormal basis with a spectral null at zero frequency. For N=2, this basis is equal to,

$$\mathbf{M}=\begin{bmatrix} \sqrt{2}/2 & \sqrt{6}/6 \\ 0 & -\sqrt{6}/3 \\ -\sqrt{2}/2 & \sqrt{6}/6 \end{bmatrix} . \tag{16}$$

Using this basis, the shaping region of **x**, denoted as $\mathcal{R}_x$, is located at the subspace $X_0+X_1+X_2=0$. We assume that the baseline region is of $\mathcal{A}_2^1$ type. After some manipulation, projection of $\mathcal{R}_x$ into the (X_0, X_1) subspace is found as,

$$\begin{cases} (X_0+0.5X_1)^2/S_0^2+0.75(X_1/S_1)^2 \leq 0.5\beta \\ -\sqrt{2} \leq (X_0+0.5X_1)/S_0 \leq \sqrt{2} \\ -\sqrt{6}/3 \leq X_1/S_1 \leq \sqrt{6}/3 \end{cases} . \tag{17}$$

Similarly, projection on (X_0, X_2) subspace is equal to,

$$\begin{cases} (X_0-X_2)^2/S_0^2+3(X_0+X_2)^2/S_2^2 \leq 2\beta \\ -\sqrt{2}/2 \leq (X_0-X_2)/S_0 \leq \sqrt{2}/2 \\ -\sqrt{6}/3 \leq (X_0+X_2)/S_2 \leq \sqrt{6}/3 \end{cases} . \tag{18}$$

Projection on (X_1, X_2) subspace is obtained by replacing X_0 by X_2 in (17).

By changing β in the rage $1 \leq \beta \leq 2$, one obtains different regions corresponding to different tradeoff points. Figures 5 and 6 show the regions corresponding to (17) and (18) for $\beta = 1.1$. Within a scale factor, the power spectrum is determined by the ratio of S_0 and S_1. Figure 7 shows the corresponding spectrum. The average energy per dimension is normalized to one. For S_0=S_1, the volume of the resulting region is maximum. This corresponds to the highest rate obtainable over dimensionality M=3 and with a spectral null at zero frequency. From Fig. 7, it is seen that the corresponding spectrum is flatter than the other two cases. By increasing the ratio of S_0 and S_1, one can tradeoff the width of the null and the rate of the code. Of course, by increasing N, we obtain higher flexibility in spectral shaping.

6. Summary

The idea of the unsymmetrical shaping is introduced. This concerns the selection of the boundary of a signal constellation which has different powers along different dimensions. This problem is formulated in terms of the conventional (symmetrical) case. Two types of the shaping region and a method for the corresponding addressing are studied. In specific, the application of the unsymmetrical shaping to the spectral shaping is studied.

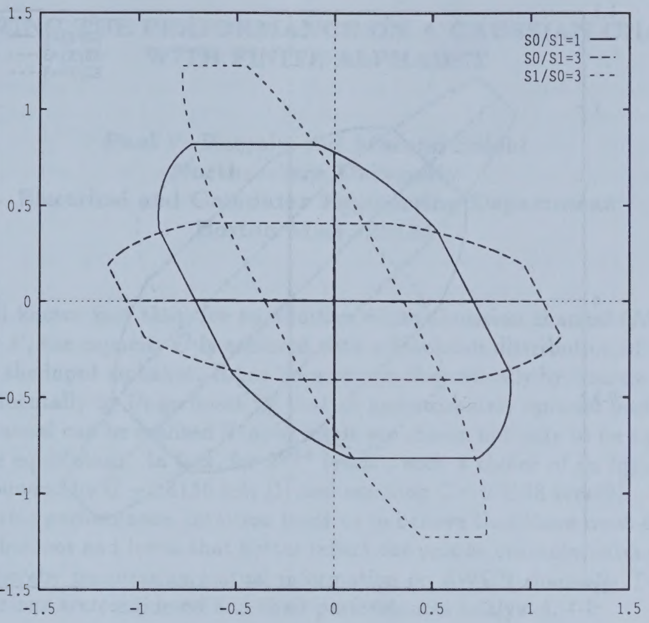


Fig. 5 Projection of the region $\mathcal{R}_x$ into the (X_0, X_1) subspace.

References

[1] A. R. Calderbank and L. H. Ozarow, "Nonequiprobable signaling on the Gaussian channel," *IEEE Trans. Inform. Theory,* vol. IT–36, pp. 726–740, July 1990.

[2] G. D. Forney and L. F. Wei, "Multidimensional constellations–Part I: Introduction, Figures of merit, and generalized cross constellations," *IEEE J. Select. Areas Commun.*, vol. SAC–7, pp. 877–892, August 1989.

[3] A. K. Khandani and P. Kabal, "Shaping multi-dimensional signal spaces: Optimum shaping, multi-stage shell mapping," Submitted to the *IEEE Trans. Inform. Theory,*.

[4] A. K. Khandani and P. Kabal, "Shaping multi-dimensional signal sets: Shell-addressed constellations," Submitted to the *IEEE Trans. Inform. Theory,*.

[5] G. D. Forney, "Multidimensional constellations-Part II: Voronoi constellations," *IEEE J. Select. Areas Commun.*, vol. SAC–7, pp. 941–958, August 1989.

[6] G. L. Cariolaro and G. P. Tronca, "Spectra of Block Coded Digital Signals," *IEEE Trans. Commun.*, vol. COM–22, pp. 1555–1563, Oct 1974.

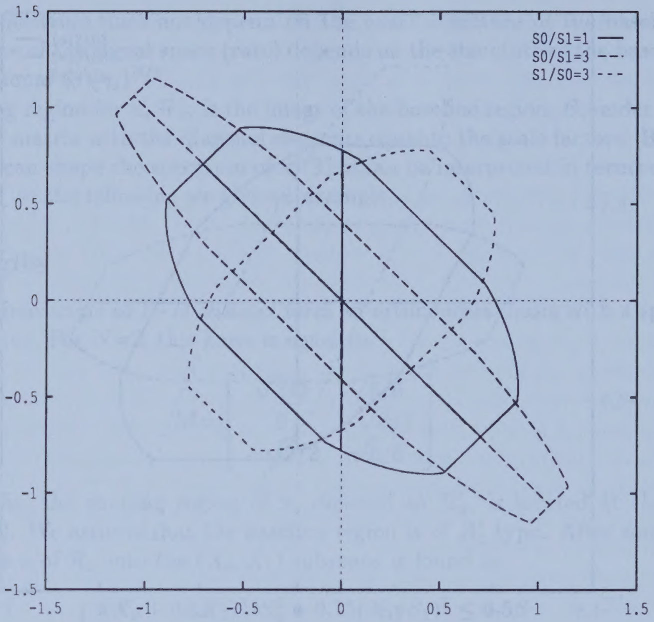


Fig. 6 Projection of the region $\mathcal{R}_x$ into the (X_0, X_2) subspace.

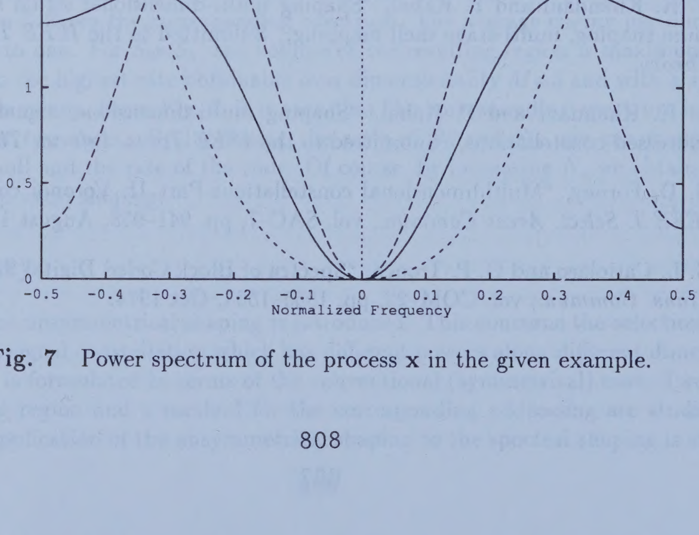


Fig. 7 Power spectrum of the process **x** in the given example.

OPTIMIZING THE PERFORMANCE ON A GAUSSIAN CHANNEL WITH FINITE ALPHABET

Paul F. Romain and Masoud Salehi
Northeastern University
Electrical and Computer Engineering Department
Boston Mass. 02115

It is a well known fact that, for an additive white Gaussian channel (AWGN) with limited power P, the capacity C is achieved with a Gaussian distribution of mean 0 and variance P on the input alphabet. It has been proven theoretically by Ozarow and Wyner [1] and experimentally by Ungerboeck [2] that an approximately optimal performance on a Gaussian channel can be realized if input levels are chosen not only to be equally likely, but also to be equidistant. In fact, for 2^{C+1} levels , such a choice of an input results in rates lower-bounded by $C - 0.4156$ bits [1] and reaching $C - 0.2158$ bits [2] . While there is no denying this performance, intuition leads us to believe that there must exist choices of input distributions and levels that better reflect the unique characteristics of Gaussian distribution thereby maximizing mutual information on AWGN channels. Two different input distributions are considered and their performances analyzed.

First, let us assume as it is commonly done that codes of equiprobable occurrence of channel input signals are of interest. That constraint in mind, the area under the probability distribution function (pdf) of a Gaussian random variable $U$of mean 0 and standard deviation σ is divided in N equal parts. The centroid of each subdivision is found. A discrete random variable X is then defined to take the values of the previously mentioned centroids with equal probability. This random variable X is used as input to the AWGN channel.

It is noteworthy to mention that X conserves the mean value of 0. However, the power of this input signal, as would be expected, comes up to be less than σ^2. Using this scheme, it is found that for $N = 2^{C+1}$, rates up to $C - 0.1463$ bits are possible. While these results seem to indicate an encouraging improvement, it is important to realize that regardless of how N is selected in function of C, capacity, even in the assumption of limiting parameter values, is never achievable.

Any time an equally likely random variable is considered as input signal, the performance of such a channel code seems to be restricted regardless of the alphabet size N. It is obvious that even in the limiting case of N reaching infinity, the above input distribution, restricted by the equiprobable occurrence of its N possible values, is never able to approach the Gaussian distribution. In fact, it can be proven that as N gets larger and larger, mutual information becomes lower-bounded by $\log N$. Consequently, another channel input has to be defined. The logical approach would be to look at mutual information as the quantity to maximize and to derive analytically if possible the input distribution that achieve such results. This derivation reveals itself to be quite tedious if not impossible. These circumstances lead us to take an intuitive look at Max's method of quantization for minimum distortion.

Going through Max's work, there is not any theoretical fact or observation that could be used to argue that a discrete random variable resulting from the quantization of a Gaussian random variable of mean 0 and standard deviation σ and used as input results in much improved performance when compared to previously discussed input alphabets. In his derivation , Max was concerned about minimizing distortion, certainly not maximizing mutual information. However, following our intuition, we consider such a channel input and decide to study and derive bounds for its performance.

Obviously, the random variable thus constructed maintains the mean value of 0 and presents, as could be predicted, a value for power which is much closer to σ^2 than previously. It is found that the performance of this choice of input distribution exhibit remarkable improvements when compared to results obtained from the cases characterized by equiprobable levels. In fact, for $N = 2^{C+1}$, rates up to $C - 0.0279$ bits are proven to be possible.

Achievable rates in bits for $N = 2^{C+1}$

Case	rate	Comments
1	$C - 0.4156$	Ozarow-Wyner's bound.
2	$C - 0.2158$	Ungerboeck's results.
3	$C - 0.1463$	Equiprobable quantized inputs.
4	$C - 0.0279$	Quantizing input based on Max's scheme.

Table 1

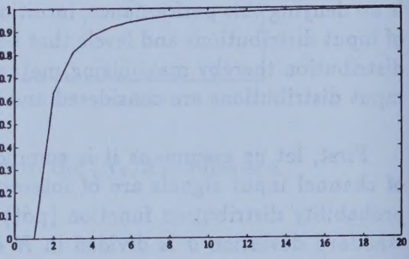


Figure 1

It can be shown that the capacity of the input quantized channel, C^*, satisfies the following bound :

$$C^* \geq C - \log(1 + P - P^2/\sigma^2)$$

where we have assumed that the noise power is equal to unity. Figure 1 shows P/σ^2 as a function of N, the number of quantization levels, when Max's quantization scheme is used. Similar plots can be obtained for other quantization schemes. From this figure, it is apparent that for large values of N, $P/\sigma^2 \approx 1$ and therefore the above bound approaches C, the capacity of the non-quantized channel as expected.

References

[1] L. H. Ozarow and A. D. Wyner,"On the Capacity of the Gaussian channel with a finite number of input levels," *IEEE Trans. Inform. Theory*,vol. IT-36, pp. 1426-1428, Nov. 1990.

[2] G. Ungerboeck,"Channel coding with multilevel/phase signals," *IEEE Trans. Inform. Theory*,vol. IT-28, pp. 55-67,Jan. 1982.

Fundamental Algorithms on Reconfigurable Meshes[†]

Stephan Olariu[‡], *James L. Schwing, and*
Jingyuan Zhang

Department of Computer Science
Old Dominion University
Norfolk, VA 23529-0162
U.S.A.

ABSTRACT

A bus system is called *reconfigurable* if it can be dynamically changed to suit computational needs. The purpose of this paper is to introduce a novel data movement technique for a two-dimensional reconfigurable mesh (also termed a 2-d PARBS). As applications we present algorithms for sorting and list ranking. More precisely, to sort a sequence of n real numbers, our sorting algorithm takes $O(\frac{\log n}{\log m})$ time on a PARBS of size $nm \times n$ with $3 \le m \le n$, and to rank a list of length n, our list ranking algorithm takes $O(\frac{\log n}{\log m})$ time on a PARBS of size $(m(n+1)+2) \times 3n$ with $2 \le m \le n$. Our results imply, in particular, that sorting n real numbers takes O(1) time on a PARBS of size $n^{1.5} \times n$, and ranking a list of length n takes O(1) time on a PARBS of size $(n^{1.5}+n^{0.5}+2) \times 3n$.

Index Terms: reconfigurable meshes, bus systems, parallel algorithms, prefix sums, sorting, list ranking, PARBS

1. Introduction

To overcome the inefficiency of long distance communications among processors, bus systems have been recently added to a number of parallel machines [2-10]. If such a bus system can be dynamically changed, under program control, to suit communication needs among processors, it is referred to as *reconfigurable*. Examples include the *bus automaton* [7], the *reconfigurable mesh* [5,6], and the *polymorphic torus* [2,3], among others.

The computational model used throughout this work is the *reconfigurable mesh.* An $m \times n$ reconfigurable mesh (also called a 2-dimensional processor array with a reconfigurable bus system or a PARBS [9,10] for short) consists of $m \times n$ identical processors positioned on a rectangular array (refer to Figure 1). The processor at (i,j), $(0 \le i \le m-1;\ 0 \le j \le n-1)$ is identified by $P(i,j)$. Every processor has 4 ports denoted by N, S, E, and W. There are also implicit *north, south, east,* and *west* directions (refer to Figure 1). In each processor, ports can be dynamically connected in pairs to suit computational needs. In the absence of these local connections, the PARBS is functionally equivalent to the mesh connected computer.

We assume a single instruction stream: in each time unit the same instruction is broadcast to all processors, which execute it and wait for the next instruction. Each instruction can consist of setting local connections (as explained later), performing an arithmetic or boolean operation, broadcasting a

---

[†] This work was supported by NASA under grant NCC1-99

[‡] Additional support by the National Science Foundation under grant CCR-8909996 is gratefully acknowledged

value on a bus, or receiving a value from a specified bus. In addition, each processor is assumed to have a constant number of registers and to be able to perform in unit time standard arithmetic and boolean operations. The regular structure of the PARBS makes it suitable for VLSI implementation. In fact, it has been argued [5] that the PARBS can be used as a universal chip capable of simulating any equivalent-area architecture without loss of time.

By adjusting the local connections within each processor several disjoint subbuses can be established. Two buses are *disjoint* if they do not share a port. We assume that the setting of local connection is destructive in the sense that setting a new pattern of connections destroys the previous one. At any given time, only one processor can broadcast a value onto a bus. Processors, if instructed to do so, read the bus. If no value is being transmitted on the bus, the read operation has no result. It is assumed [5,6] that communications along buses take O(1) time. This seems to be a reasonable assumption in the light of recent experiments with the YUPPIE system [4].

A number of problems have been solved in O(1) time on PARBS. Very recently, Wang et al [10] have proposed O(1) algorithms for the transitive closure and some related graph problems; Wang, Chen, and Lin [9] have proposed an O(1) sorting algorithm on a PARBS system. Their computational model is a 3-d PARBS containing two kinds of processors: some of the processors have four ports, while others have six ports. In spite of its constant time performance, their algorithm uses $O(n^3)$ processors.

The purpose of this paper is to introduce a data movement technique on the PARBS. This technique is interesting in its own right: it involves using taking advantage of the reconfigurability of the PARBS to perform modulo m arithmetic very efficiently. In addition, the technique can be exploited to obtain fast algorithms for a number of problems. To begin, we show that the problem of computing the prefix sums of a binary sequence of size n can be solved in time $O(\frac{\log n}{\log m})$ on a PARBS of size $m \times n$ with $3 \le m \le n$. Using this algorithm as a subroutine we can sort n real numbers in $O(\frac{\log n}{\log m})$ time on a PARBS of size $nm \times n$ with $3 \le m \le n$. In particular, for $m = n^{0.5}$ we obtain an O(1) time sorting algorithm on a PARBS of size $n^{1.5} \times n$. Our algorithm uses a two-dimensional PARBS with a uniform structure (only four ports per processor) and needs significantly fewer processors than the algorithm in [9].

The problem of *list ranking* is to determine in parallel the rank of every element in a given linked list, that is, for each element, the number of elements preceding it in the list. List ranking has turned out to be one of the fundamental techniques in parallel processing, playing a crucial role in a vast array of important parallel algorithms [1]. To the best of our knowledge, no list ranking algorithm has been reported for the PARBS. One of the goals of this paper is to present such an algorithm. Specifically, a linked list of size n can be ranked in $O(\frac{\log n}{\log m})$ time on a PARBS of size $(m(n+1)+2) \times 3n$ with $2 \le m \le n$. Again, this result implies, in particular, that ranking a list of length n takes O(1) time on a PARBS of size $(n^{1.5}+n^{0.5}+2) \times 3n$.

This paper is organized as follows: Section 2 illustrates our data movement technique as applied to the problem of computing the prefix sums of a binary sequence; Section 3 discusses our sorting algorithm; finally, Section 4 deals with the list ranking algorithm.

2. Prefix sums of a binary sequence

Given a binary sequence $b_0, b_1, \ldots, b_{n-1}$, where b_j $(0 \le j \le n-1)$ is 0 or 1, the problem is to find $z_j = b_0 + \cdots + b_j$ for $0 \le j \le n-1$. We first show how to solve this problem in constant time on an $(n+1) \times n$ PARBS. We then extend the idea to solve the problem in $O(\frac{\log n}{\log m})$ time on a PARBS of size $(m+1) \times 2n$ with $2 \le m \le n$. Finally, we show how to reduce the size of the PARBS to $m \times n$ with $3 \le m \le n$ while keeping the running time in $O(\frac{\log n}{\log m})$.

Computing $z_j = b_0 + \cdots + b_j$ for $0 \le j \le n-1$ is easy on an $(n+1) \times n$ PARBS, with b_j being stored by $P(n,j)$. If b_j is 0 then all processors $P(i,j)$ with $0 \le i \le n$ connect their W and E ports; if b_j is 1 then all processors $P(i,j)$ with $0 \le i \le n$ connect their W and S ports and their N and E ports. Now $P(0,0)$ broadcasts a "#" signal from its W port. It is easy to confirm that for $0 \le j \le n-1$, the row number of the unique processor that receives "#" from its E port, equals $z_j = b_0 + \cdots + b_j$. See Figure 2 for an example: the binary sequence is (1,0,1,1), and the highlighted bus is the one on which "#" is broadcast.

In case the number of rows in the PARBS is $m+1$, less than $n+1$, we may not have sufficient height to store the prefix sums as described above. We will therefore adapt the algorithm by performing a "wrap-around" technique which emulates modulo m arithmetic. Our algorithm involves a PARBS of size $(m+1)\times 2n$ where we assume the input has been distributed to the processors at the bottom of the even columns, $P(m,2j)$.

We now outline the basic idea of our algorithm. Commence by setting switches in even columns as described above. For the odd columns, except for the top and bottom rows, switches are set as a cross (i.e connect N to S, and E to W, with no intersection). This allows east-west traffic to pass as in the previous algorithm, while allowing information to be passed from the bottom processor of an odd column to the top one. Switches on the bottom row are set to pass information from $P(m,2j)$ to $P(m,2j+1)$. Similarly, switches in the top row are set to pass information from $P(0,2j+1)$ to $P(0,2j+2)$. Figure 3 shows such an example for $m=2$, $n=4$, and input (1,0,1,1).

If one now applies the previous algorithm and calls the resulting values y_j^1 (i.e. for every odd column $2j+1$, y_j^1 is the row number of the unique processor that receives "#" from its E port), it is clear that

$$y_j^1 = z_j \bmod m.$$

Next, mark each column pair in which a wrap-around occurred. Determine a new set of binary values, b_j^1, by assigning 1 to the marked even column processors and 0 to the other even column processors of row m.

It is worthwhile to note that

$$z_j = (b_0^1 + \cdots + b_j^1)m + y_j^1.$$

The objective now becomes the computation of the partial sum of the sequence of b_j^1. It is further interesting to note that the number of times 1 occurs in the derived sequence b_j^1 is a factor of m times less than in the original sequence of b_j.

This process can be carried out iteratively, say t times. The final time there will be less than m 1's and therefore no wrap-around. It is obvious from the above construction that

$$z_j = (\cdots (((0)m + y_j^t)m + y_j^{t-1}) \cdots)m + y_j^1.$$

Expanding we get

$$z_j = y_j^t m^{t-1} + \cdots + y_j^2 m + y_j^1$$

a base m expansion of z_j. Equation (4) yields the final insight into this algorithm. Namely, for the first iteration set

$$z_j^1 \leftarrow y_j^1$$

and thereafter set

$$z_j^k \leftarrow y_j^k m^{k-1} + z_j^{k-1}.$$

The following are the details of our implementation. For ease of understanding variables will be superscripted in the algorithm. From the implementation point of view, only one set of variables is needed which are updated appropriately during each iteration.

```
Algorithm Parallel-Prefix-Sums;
{Model:  a PARBS of size $(m+1)\times 2n$ with $2\le m\le n$;
 Input:  a binary sequence $b_0, b_1, \ldots, b_{n-1}$, with $b_j$ stored by $P(m,2j)$ $(0\le j\le n-1)$;
 Output: $z_j=b_0+\cdots+b_j$ $(0\le j\le n-1)$ stored in $P(m,2j)$;}
Step 0   $k \leftarrow 0$;
         $h \leftarrow 1$;
         $P(m,2j)$ $(0 \le j \le n-1)$ sets $z_j^k \leftarrow 0$ and $b_j^k \leftarrow b_j$;
Step 1   while OR($b_0^k, b_1^k, \ldots, b_{n-1}^k$)$\neq 0$ do
    Step 1.1  {$P(m,2j)$ $(0 \le j \le n-1)$ broadcasts $b_j^k$ to all processors in columns $2j$}
              $P(i,2j)$ $(1 \le i \le m-1, 0 \le j \le n-1)$ connect $N$ and $S$;
              $P(m,2j)$ $(0 \le j \le n-1)$ writes $b_j^k$ to $N$;
              $P(i,2j)$ $(0 \le i \le m-1, 0 \le j \le n-1)$ reads $b_j^k$ from $S$;
    Step 1.2  {$P(i,2j)$ $(0 \le i \le m, 0 \le j \le n-1)$ sends $b_j^k$ to $P(i,2j+1)$;}
              $P(i,2j)$ $(0 \le i \le m, 0 \le j \le n-1)$ writes $b_j^k$ to $E$;
              $P(i,2j+1)$ $(0 \le i \le m, 0 \le j \le n-1)$ reads $b_j^k$ from $W$;
    Step 1.3  {establish subbuses}
              for all j $(0 \le j \le n-1)$ do the following:
                  if $b_j^k=1$ then
                      $P(0,2j)$ connects $W$ and $S$;
                      $P(i,2j)$ $(1 \le i \le m-1)$ connects $W$ and $S$, and $N$ and $E$;
                      $P(m,2j)$ connects $N$ and $E$;
                      $P(m,2j+1)$ connects $W$ and $N$;
                      $P(i,2j+1)$ $(1 \le i \le m-1)$ connects $N$ and $S$, and $W$ and $E$;
                      $P(0,2j+1)$ connects $S$ and $E$
                  else
                      $P(i,2j)$ and $P(i,2j+1)$ $(0 \le i \le m-1)$ connects $W$ and $E$
    Step 1.4  {generate the next binary sequence based on where the wrap-around occurred}
              $P(0,0)$ broadcasts a '#' signal from $W$;
              $P(i,2j)$ $(0 \le i \le m, 0 \le j \le n-1)$ reads from $E$;
              {notice that exactly one processor in column $2j$ receives '#' from $E$}
              if $P(m,2j)$ receives the signal "#" then
                  $b_j^{k+1} \leftarrow 1$;
              else
                  $b_j^{k+1} \leftarrow 0$;
              every processor $P(i,2j)$ that receives the signal "#" marks itself;
    Step 1.5  {define $y_j^{k+1}$}
              $P(i,2j)$ $(1 \le i \le m, 0 \le j \le n-1)$ connects $N$ and $S$;
              every marked processor $P(i,2j)$ broadcasts $i$ from its $S$ port;
              $P(m,2j)$ reads its $N$ port and sets $y_j^{k+1} \leftarrow$ data received;
              if $y_j^{k+1}=m$ then $y_j^{k+1} \leftarrow 0$;
    Step 1.6  {define $z_j^{k+1}$}
              $P(m,2j)$ $(0 \le j \le n-1)$ sets $z_j^{k+1} \leftarrow h \times y_j^{k+1} + z_j^k$;
    Step 1.7  {every processor $P(m,2j)$ updates its own copy of $k$ and $h$}
              $k \leftarrow k+1$;
              $h \leftarrow m \times h$;
Step 2   for all $j$ $(0 \le j \le n-1)$ $z_j \leftarrow z_j^{k-1}$;
```

Theorem 1. Algorithm Parallel-Prefix-Sums correctly computes the prefix sums of a binary sequence $b_0, b_1, \ldots, b_{n-1}$, where b_j is stored by processor $P(m,2j)$ $(0 \le j \le n-1)$, in $O(\frac{\log n}{\log m})$ time on a PARBS of size $(m+1) \times 2n$ with $2 \le m \le n$. □

Corollary 1.1. The prefix sums of a binary sequence $b_0, b_1, \ldots, b_{n-1}$ can be computed in $O(\frac{\log n}{\log m})$ time on a PARBS of size $m \times n$ with $3 \le m \le n$.

Proof. The result can be derived from the observation that the prefix sums can be computed as above by using emulation of modulo $(m-1)$ arithmetic on a PARBS of size $m \times n$ $(3 \le m \le n)$ and combining the prefix sums of its even subsequence and the prefix sums of its odd subsequence. □

3. Sorting

Our sorting algorithm is based on enumeration sort. Assume the numbers to be sorted are a_0, $a_2, \ldots, a_{n-1}$. The idea is to compute for every a_i $(0 \le i \le n-1)$, its rank $r(i)$, that is, the number of elements that precede it in the sorted sequence.

First, we compare every a_i with all a_j. For every such pair, record a 1 if $a_i > a_j$, and a 0 otherwise. What results is, for every i $(0 \le i \le n-1)$, a binary sequence of length n. The number of 1's in this sequence stands for the rank of a_i. Clearly, to compute the rank of a_i $(0 \le i \le n-1)$ we only need evaluate the sum of the corresponding binary sequence. For this purpose we use the (prefix) sums algorithm developed in the Section 2.

```
Algorithm Parallel-Sorting;
{Model:   a PARBS of size $mn \times n$ with $3 \le m \le n$;
 Input:   a sequence of numbers $a_0, a_1, \ldots, a_{n-1}$ with $a_j$ $(0 \le j \le n-1)$ stored by $P(0,j)$;
 Output:  the sequence $a_0, a_1, \ldots, a_{n-1}$ sorted in ascending order, stored by $P(0,0)$,
          $P(0,1), \ldots, P(0,n-1)$}

Step 1    {compare $a_i$ with $a_j$ $(0 \le i \le n-1, 1 \le j \le n-1)$}
          Step 1.1  processor $P(0,j)$ $(0 \le j \le n-1)$ broadcasts $a_j$ to the whole column $j$;
                    processor $P(im,i)$ $(0 \le i \le n-1)$ broadcasts $a_i$ to the whole row $im$;
          Step 1.2  {now every processor $P(im,j)$ knows $a_i$ and $a_j$}
                    if ($a_i > a_j$ or $a_i = a_j$ and $i > j$) then
                        $b_{ij} \leftarrow 1$
                    else
                        $b_{ij} \leftarrow 0$;
Step 2    invoke Parallel-Prefix-Sums to compute the sum $b_{i0} + b_{i1} + \cdots + b_{in-1}$ (i.e. the rank of $a_i$) in
          the sub-PARBS of size $m \times n$ starting at row $im$ and ending at row $im+m-1$.
          {the result, called $r(i)$, is stored in $P(im,n-1)$}
Step 3    $P(im,n-1)$ broadcasts $r(i)$ to $P(im,r(i))$ $(0 \le i \le n-1)$;
          $P(im,r(i))$ broadcasts $a_i$ to $P(0,r(i))$;
```

Theorem 2. Algorithm Parallel-Sorting correctly sorts a sequence of n reals in $O(\frac{\log n}{\log m})$ time on a PARBS of size $nm \times n$ with $3 \le m \le n$.

Corollary 2.1. A sequence of n reals can be sorted in O(1) time on a PARBS of size $n^{1.5} \times n$.

4. List ranking

Given a linked list of length n, the problem is to compute for every element its rank in the list, that is, the number of elements preceding it in the list. In this section we first develop a list ranking algorithm running in O(1) time on a $(n^2-1) \times 2n$ PARBS. Then, we propose a list ranking algorithm that runs in $O(\frac{\log n}{\log m})$ time using a PARBS of size $(m(n+1)+2) \times 3n$ with $2 \le m \le n$.

We assume a linked list stored in an unordered array $A[0..n-1]$, with each element except for the last storing the index of its successor in the linked list. For simplicity, link$(i) = j$ (or $i \rightarrow j$), denotes the fact that $A[j]$ is the successor of $A[i]$ in the linked list.

We begin by describing the algorithm for computing the rank of every element in the list on an $(n^2-1)\times 2n$ PARBS. We assume that every $A[j]$ is stored by $P(0,2j)$. We view this PARBS as being partitioned into $n-1$ subPARBS of $n+1$ rows each. The algorithm proceeds identically in each subPARBS. For simplicity of exposition, we only refer to the subPARBS consisting of rows 0 through n. If $A[j]$ points to $A[k]$ then a subbus will be established as follows:

- vertically from $P(0,2j)$ to $P(j,2j)$;
- horizontally from $P(j,2j)$ to $P(j,2k+1)$;
- vertically from $P(j,2k+1)$ to $P(n,2k+1)$;
- horizontally from $P(n,2k+1)$ to $P(n,2k)$.
- $P(0,2j)$ connects S with N and $P(n,2k)$ connects E with S.

It is easy to confirm that as a result, each subPARBS features exactly $n-1$ disjoint buses, each corresponding to a relationship link(j)=k. Note that the bus corresponding to link(j)=k begins with $N(0,2j)$ and ends with $S(n,2k)$.

The global effect is to partition the original PARBS into a set of disjoint buses. We are only interested in the bus starting at $N(0,2h)$ with $A[h]$ the head of the list. Now processor $P(0,2h)$ broadcasts a "#" signal on this bus from its N port; it is easy to confirm that for $0\leq j\leq n-1$, processor $P(r(n+1)-1,2j)$ $(r\geq 1)$ receives "#" from its S port if and only if the rank of $A[j]$ is r. Figure 4 gives an example with n=4 and the list 1 → 3 → 0 → 2. The highlighted bus is the one on which the signal "#" is broadcast.

Notice that if the PARBS has strictly fewer rows, the above algorithm can not be applied directly. The idea of our list ranking algorithm is to reduce the original problem to that of ranking shorter and shorter lists until the list at hand is sufficiently short for the above algorithm to be applied directly.

More precisely, we assume a PARBS of size $(m(n+1)+2)\times 3n$. As before, we think of this PARBS as being partitioned into m subPARBS of $n+1$ rows each. In addition, there are two special rows, namely row 0 and row $m(n+1)+1$ as well as n special columns whose purpose will be described later.

An outline of our algorithm follows. Given a linked list L of length n we construct a new linked list L' by extracting from L every m-th element starting with the head of the list, i.e. the element whose rank in L is a multiple of m. With each $A[j]$ associate an ordered pair ($A[j']$,offset(j)) such that

- $A[j']=A[j]$ if $A[j]$ belongs to L'; otherwise $A[j']$ is the closest element preceding $A[j]$ in L and belongs to L';
- offset(j) stands for the number of links that have to be traversed in L to get from $A[j']$ to $A[j]$.

Assume that we can recursively rank L', it is easy to see that the rank of $A[j]$ in L equals $m\times$(rank of $A[j']$ in L')+offset(j).

Note that the length of L' is $\left\lceil \frac{n}{m} \right\rceil$. Consequently, the depth of the recursion described above is bounded by $c+1$, where $c=\left\lfloor \frac{\log n}{\log m} \right\rfloor$.

For the purpose of implementing this idea on a PARBS we shall use a similar iterative method instead of recursion. We proceed in the following two stages.

Stage 1. We start out with the original list L that is referred to as L_0. In every iteration t $(t\geq 1)$ of Stage 1, a new list L_t is created by retaining from L_{t-1} every m-th element, and eliminating the others. Each element $A[j]$ eliminated in the t-th iteration remembers an ordered pair (t,offset(j)) with offset(j) representing the number of links to traverse in L_{t-1} to get from $A[j']$ that is the closest element preceding $A[j]$ in L_{t-1} and belongs to L_t to $A[j]$ itself.

The first Stage ends when we get the list L_{c+1} with c as above, a list with just one element, namely the head of the original list. Furthermore, we take the rank of the head in L_{c+1} to be 0, as it should.

Stage 2. In this Stage we traverse the set of (implicit) lists L_j $(0\leq j\leq c+1)$ in reverse order and use the rank of every element in L_t to compute the rank of every element in L_{t-1}. Here, we use the formula stated before, i.e. rank of $A[j]$ in L_{t-1} equals $m\times$(rank of $A[j']$ in L_t)+offset(j).

In Stage 1, we need to construct the new list L_t out of the old list L_{t-1}. In a PARBS of size $(m(n+1)+2)\times 3n$, by using the similar technique above, we can readily construct a bus corresponding to L_{t-1}, which starts at row 1 and proceeds to row $m(n+1)$ and then wraps around coming back to row 1. Rows 0 and $m(n+1)+1$ as well as columns $3j+2$ $(0\le j\le n-1)$ are used for this purpose only. Now extracting the list L_t involves sending a signal along this bus and retaining every element in row 1 that receives the signal. To implement this idea we rely on the following algorithm to construct a set of buses. The details follow.

Algorithm Convert-List-to-Bus1(L_t);

{Model: a PARBS of size $(m(n+1)+2)\times 3n$;

Input: a linked list L_t with each element stored in a marked $P(0,3j)$ $j\in\{0,1,\ldots,n-1\}$; the head of the list is in $P(0,3h)$; every marked $P(0,3j)$ stores link(j);

Output: the bus starting at port $S(0,3h)$ and corresponding to L_t;}

```
for all $k$ $(0\le k\le m-1)$ do
    for all $j$ $(0\le j\le n-1)$ such that $P(0,3j)$ is marked and link($j$)$\ne$nil
        {The following bus established corresponds to $j\rightarrow$ link($j$).}
        establish a bus starting from $P((n+1)k+1,3j)$
        first to $P((n+1)k+j+1,3j)$ vertically,
        then to $P((n+1)k+j+1,3\text{link}(j)+1)$ horizontally,
        then to $P((n+1)k+n+1,3\text{link}(j)+1)$ vertically,
        finally to $P((n+1)k+n+1,3\text{link}(j))$ horizontally;
        $P((n+1)k+1,3j)$ connects $S$ with $N$;
        $P((n+1)k+n+1,3\text{link}(j))$ connects $E$ with $S$;
for all $j$ $(0\le j\le n-1)$ such that $P(0,3j)$ is marked and link($j$)$\ne$nil
    {The following bus established is for wrapping around.}
    establish a bus starting from $P((n+1)m+1,3\text{link}(j))$,
    first to $P((n+1)m+1, 3\text{link}(j)+2)$ horizontally;
    then to $P(0, 3\text{link}(j)+2)$ vertically,
    finally to $P(0, 3\text{link}(j))$ horizontally;
    $P((n+1)m+1, 3\text{link}(j))$ connects $N$ with $E$;
    $P(0, 3\text{link}(j))$ connects $E$ with $S$;
```

Observation 1. the bus starting at $S(0,3h)$ passes through $S(k(n+1),3j)$, $k=r$ mod m, where r is the rank in L_t of the element stored by $P(0,3j)$.

An example to clarify Observation 1 is given in Figure 5, where $m=2$, $n=4$ and the linked list is $1\rightarrow 3\rightarrow 0\rightarrow 2$.

In stage 2, we use the original linked list L and the iteration number to determine the sublist L_t. This is the reason for retaining the iteration number for every eliminated element in Stage 1 and for making a copy of the original list initially. At the beginning of Stage 2, we use the following algorithm Convert-list-to-Bus2 to convert a linked list to a corresponding bus that keeps the order of the linked list.

Algorithm Convert-List-to-Bus2(L);

{Model: a PARBS of size $(n+1)\times 3n$.

Input: a linked list L of length n with the head $A[h]$; Each element $A[j]$ $(0\le j\le n-1)$ is stored in $P(0,3j)$; The successor of element $A[j]$, link(j), is also stored in $P(0,3j)$;

Output: the bus starting at $S(0,3h)$ and corresponding to L;}

```
for all $j$ ($0 \le j \le n-1$) such that link($j$)$\ne$nil
        establish a bus starting from $P(0,3j)$
        first to $P(j+1,3j)$ vertically,
        then to $P(j+1,3\text{link}(j)+1)$ horizontally,
        finally to $P(0,3\text{link}(j)+1)$ vertically;
        $P(0, 3\text{link}(j)+1)$ connects $S$ with $W$;
        $P(0, 3\text{link}(j))$ connects $S$ with $E$;
```

Observation 2 With a linked list L as input, Convert-List-to-Bus2 establishes in O(1) time a bus beginning with $S(0,3h)$ and satisfying the following:

- the bus passes through all $P(0,3j)$ ($0 \le j \le n-1$);
- if $A[j]$ precedes $A[k]$ in L, $P(0,3j)$ is traversed before $P(0,3k)$ if the bus is traversed from $P(0,3h)$.

An example is given to clarify how the bus is established in Figure 6, where $n=4$ and the linked list is $1 \to 3 \to 0 \to 2$.

Now we are in the position to give the algorithm.

```
Algorithm Parallel-List-Ranking;
{Model:   a PARBS of size $(m(n+1)+2) \times 3n$;
 Input:   a linked list $L$ of length $n$ with the head $A[h]$; Each element $A[j]$ ($0 \le j \le n-1$) is stored in
          $P(0,3j)$; The successor of element $A[j]$, link($j$), is also stored in $P(0,3j)$;
 Output:  rank($j$) ($0 \le j \le n-1$), the rank of $A[j]$ in the linked list $L$, which is stored in $P(0,3j)$.}

Step 0    {initialization}
          $t \leftarrow 0$;
          $P(0,3j)$ ($0 \le j \le n-1$) does the following:
                link'($j$)$\leftarrow$link($j$); {make a copy $L'$ of $L$}
                marks itself;
Step 1    while (there are more than one marked $P(0,3j)$ ($j \in \{0,1,...,n-1\}$)) do
          Step 1.1   {convert the linked list $L_t$ to a bus.}
                     Convert-List-to-Bus1($L_t$);
          Step 1.2   {compute $L_{t+1}$ from $L_t$ and record (iter($j$), offset($j$)) for all the eliminated ele-
                     ments.}
                     every marked processor unmarks itself;
                     $P(0,3h)$ broadcasts a signal '#' from $S$;
                     $P((n+1)k,3j)$ ($0 \le j \le n-1$, $1 \le k \le m-1$) that has received "#" from $S$ does:
                           (iter($j$), offset($j$)) $\leftarrow$ ($t$, $k$);
                           mark itself "special"; {The corresponding element is eliminated.}
                     $P(0,3j)$ ($0 \le j < n-1$) that has received "#" from $S$ does:
                           mark itself; {The corresponding element is retained.}
                           {for every $i, j$ with link($i$)=$j$ in $L_{t+1}$, create a subbus starting from
                           $E(0,3j)$ and ending at $S(0,3i)$; every $P(0,3j)$ ($j \ne h$) broadcasts $j$ along
                           the appropriate subbus; $P(0,3i)$ reads this information from its port $S$}
                           $P(0,3j)$ splits the bus by disconnecting $E$ and $S$;
                           $P(0,3j)$ ($j \ne$ h) broadcasts $j$ from its port $E$;
                           if $P(0,3j)$ received a value from its port $S$ then
                                 link($j$) $\leftarrow$ the value received;
                           else
                link($j$) ← nil;
        Step 1.3  {update $t$}
                  $t \leftarrow t+1$;
Step 2  {prepare for computing rank}
        every "special" $P((n+1)k,3j)$ ($0 \le j \le n-1$, $1 \le k \le m-1$) sends (iter($j$),offset($j$)) to $P(0,3j)$;
        $P(0,3h)$ does:
            (iter($h$),offset($h$))←($t$, 0);
            rank($h$)←0;
        {update the value of $t$ to mimic the corresponding iteration number in Stage 1}
        $t \leftarrow t-1$;
        Convert-List-to-Bus2($L'$); {convert the original linked list $L'$ to a bus}
Step 3  {compute the rank.}
        while ($t \ge 0$) do
            {We use the ranks of the elements in $L_{t+1}$ to compute the ranks of the elements in $L_t$;
            note that the elements in $L_{t+1}$ have iter($j$)>$t$}
            every $P(0,3j)$ with iter($j$)>$t$ splits the bus by disconnecting $E$ and $S$ and broadcasts
            rank($j$) from $S$;
            every $P(0,3j)$ with iter($j$)≥$t$ does:
                reads its port $S$;
                rank($j$)←$m$×(the value received)+offset($j$);
                offset($j$)←0;
            $t \leftarrow t-1$;
```

Theorem 3. Algorithm Parallel-List-Ranking correctly computes the rank of each element in the linked list of length n in $O(\frac{\log n}{\log m})$ time on a PARBS of size $(m(n+1)+2)\times 3n$. □

Corollary 3.1. The problem of ranking a linked list of length n can be solved in O(1) time on a PARBS of size $(n^{1.5}+n^{0.5}+2)\times 3n$. □

References

1. S. G. Akl, *The design and analysis of parallel algorithms*, Prentice-Hall, 1989.

2. H. Li and M. Maresca, Polymorphic-torus network, *IEEE Transactions on Computers*, vol. C-38, no. 9, (1989) 1345-1351.

3. H. Li and M. Maresca, Polymorphic-torus architecture for computer vision, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 11, no. 3, (1989) 233-243.

4. M. Maresca and H. Li, Connection autonomy and SIMD computers: a VLSI implementation, *Journal of Parallel and Distributed Computing*, vol. 7 (1989) 302-320.

5. R. Miller, V. K. Prasanna-Kumar, D. Reisis, and Q. F. Stout, Meshes with reconfigurable buses, *Proceedings of the International Conference on Parallel Processing*, vol. 1, (1988) 205-208.

6. R. Miller, V. K. Prasanna-Kumar, D. Reisis, and Q. F. Stout, Data movement operations and applications on Reconfigurable VLSI arrays, *Proceedings of the fifth MIT Conference on Advanced Research in VLSI*, (1988) 163-178.

7. J. Rothstein, Bus automata, brains, and mental models, *IEEE Trans. on Systems Man Cybernetics* 18, (1988).

8. Q. F. Stout, Meshes with Multiple Buses, *Proc. 27th IEEE Symp. on the Foundations of Comp. Science*, (1986) 264-273.

9. B. F. Wang, G.H. Chen, and F.C. Lin, Constant Time Sorting on a processing array with a reconfigurable bus system, *Information Processing Letters*, vol. 34, no. 4, pp. 187-192, (1990).

10. B. F. Wang, C. J. Lu, and G. H. Chen, Constant Time Algorithms for the Transitive Closure Problem and its Applications, *Proceedings of the International Conference on Parallel Processing*, 3, (1990) 52-59.

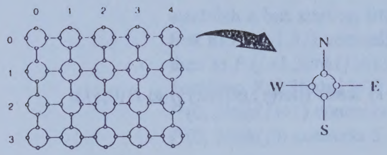

0
1
2
3
4
0
1
2
3
N
W
E
S

Figure 1. A 4x5 PARBS

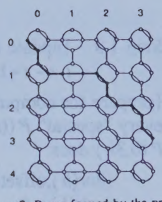


Figure 2. Buses formed by the prefix sums algorithm for 1 0 1 1

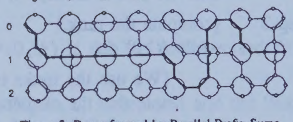

0
1
2
3
4
5
6
7
0
1
2

Figure 3. Buses formed by Parallel-Prefix-Sums for 1 0 1 1

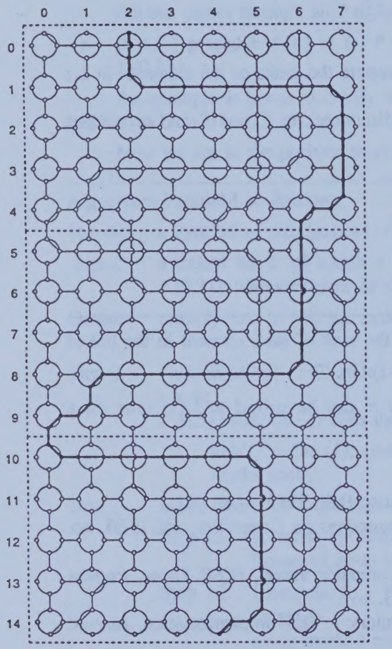

0
1
2
3
4
5
6
7
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14

Figure 4. Buses formed by the list ranking algorithm for 1→3→0→2

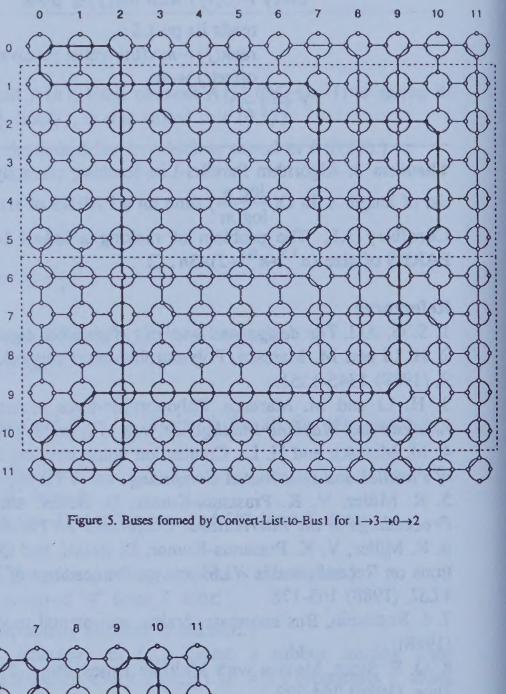

0
1
2
3
4
5
6
7
8
9
10
11
0
1
2
3
4
5
6
7
8
9
10
11

Figure 5. Buses formed by Convert-List-to-Bus1 for 1→3→0→2

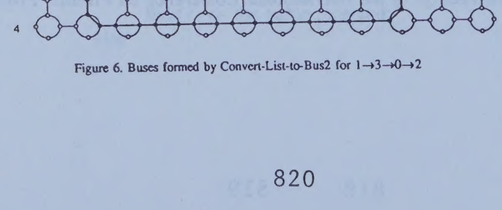

0
1
2
3
4
5
6
7
8
9
10
11
0
1
2
3
4

Figure 6. Buses formed by Convert-List-to-Bus2 for 1→3→0→2

Integer Problems on Reconfigurable Meshes, with Applications[†]

Stephan Olariu[‡]*, James L. Schwing, and Jingyuan Zhang*

Department of Computer Science
Old Dominion University
Norfolk, VA 23529-0162
U.S.A.

ABSTRACT

Problems with solutions based upon integer computations are collectively referred to as integer problems. Such problems occur routinely in pattern recognition, image processing, graph theory, and query processing. The purpose of this paper is to present constant time algorithms for computing the prefix sums of an integer sequence, and for solving the packing problem on reconfigurable meshes. These two algorithms afford us efficient solutions to a wide variety of problems: we obtain constant time solutions to integer sorting, converting between the edge-list and the adjacency list representation of a graph, converting between different representations of a rooted tree; we also obtain an O(loglog$n$) algorithm to compute the perimeter and the area of a binary image of size $n \times n$.

Index Terms: reconfigurable meshes, bus systems, integer sorting, prefix sums, graph, image processing, pattern analysis

1. Introduction

To overcome the inefficiency of long distance communications among processors, bus systems have been recently added to a number of parallel machines [2,3,5,6,8,9]. If such a bus system can be dynamically changed, under program control, to suit communication needs among processors, it is referred to as *reconfigurable.* Examples include the *bus automaton* [8], the *reconfigurable mesh* [5,6], and the *polymorphic torus* [2,3].

The computational model used throughout this work is the *reconfigurable mesh.*[*] An $m \times n$ reconfigurable mesh consists of $m \times n$ identical processors positioned on a rectangular array (refer to Figure 1). The processor located at (i,j), $(0 \le i \le m-1;\ 0 \le j \le n-1)$ is referred to as $P(i,j)$. Every processor has 4 ports denoted by N, S, E, and W. There are also implicit *north, south, east,* and *west* directions. In each processor, ports can be dynamically connected in pairs to suit computational needs. Our computational model only allows two connection to be set in each processor. Furthermore, these two connection must involve disjoint pairs of ports (see Figure 2). In the absence of these local connections, the reconfigurable mesh is functionally equivalent to the mesh connected computer.

To make the model realistic, we assume that the processing elements have a small number of registers of O(log n) bits and a very basic instruction set; a processor can perform in unit time standard

---

[†] This work was supported by NASA under grant NCC1-99

[‡] Additional support by the National Science Foundation under grant CCR-8909996 is gratefully acknowledged

[*] when no confusion is possible a reconfigurable mesh will be referred to simply as a mesh.

arithmetic and boolean operations. More sophisticated operations including computing floor and modulo computation as specifically excluded. To anticipate, we show that these operations can, in fact, be added to the repertoire with very little overhead, thus setting the stage for a very powerful computing environment. We assume a SIMD model: in each time unit the same instruction is broadcast to all processors, which execute it and wait for the next instruction. Each instruction can consist of setting local connections, performing an arithmetic or boolean operation, broadcasting a value on a bus, or receiving a value from a specified bus. The regular structure of the reconfigurable mesh makes it suitable for VLSI implementation [5,6]. In fact, it has been argued [6] that the reconfigurable mesh can be used as a universal chip capable of simulating any equivalent-area architecture without loss of time.

By adjusting the local connections within each processor several subbuses can be established. We assume that the setting of local connection is destructive in the sense that setting a new pattern of connections destroys the previous one. At any given time, only one processor can broadcast a value onto a bus. Processors, if instructed to do so, read the bus. If no value is being transmitted on the bus, the read operation has no result. It is assumed [4,5,10,11] that communications along buses take O(1) time. This seems to be a reasonable assumption in the light of recent experiments with the YUPPIE system [4].

Problems in many practical applications can be reduced to computations on integer values. Such problems are collectively referred to as *integer* problems. Although, in principle, we could solve an integer problem by bringing to bear general-purpose algorithms, the hope is that more efficient algorithms can be devised for handle these problems more efficiently. The purpose of this paper is to propose a number of efficient algorithms for integer problems on reconfigurable meshes.

To begin, we present a constant time algorithm to compute the prefix sums of a sequence of n integers ranging from 0 to $n-1$ on an $n \times n$ reconfigurable mesh. As we are about to point out, the range of the integers to be processed by both sorting and prefix sum algorithm can be extended to n^c for arbitrary c; in this case the complexity of our algorithms is $O(c)$ time using the same size reconfigurable mesh.

As an application of the integer prefix sums algorithm, we propose an efficient solution to the following problem referred to as the *packing problem*. Given an n-element set X and a collection of m $(m \leq n)$ predicates such that every element of X satisfies at most one predicate, the packing problem asks for a partion of X such that the elements of X satisfying the same predicate occur in the same subset of the partition. Likewise, the elements of X that satisfy no predicate belong to a single subset of the partition.

We show that our integer prefix sums algorithm yields an efficient algorithm to solve the packing problem. Specifically, we outline an algorithm that solves the packing problem in O(1) time on a reconfigurable mesh of size $n \times n$.

As it turns out, the solution to the packing problem can be used to devise a constant time sorting algorithm. We show that sorting n integers in the range 0 to n^c can be solved in $O(c)$ time on a reconfigurable mesh of size $n \times n$. To our best knowledge, this is the first instance of such an algorithm reported in the literature.

Packing and prefix sums of integer sequence have far-reaching applications. Among them, we present a constant time algorithm to convert an edge-list representation of a graph to an adjacent-list representation, a constant time algorithm to convert a parent-pointer representation of a rooted tree to an standard form, and an $O(\log\log n)$ algorithm to compute the perimeter and the area of a binary image. To the best of our knowledge, this is the first time such results are reported in the literature.

The paper is organized as follows: Section 2 provides a set of tools that will be used in the following sections; Section 3 presents an algorithm to compute the prefix sums of a sequence of integers; Section 4 discusses the packing problem; Section 5 discusses a number of applications; finally, Section 7 summarizes the results and proposes a number of open problems.

2. Preliminaries

One of the most useful techniques in designing parallel algorithms on a reconfigurable mesh of size $n \times n$ involves partitioning the original mesh into $\sqrt{n}$ submeshes of size $\sqrt{n} \times n$ each[*]. At times, it may also be necessary to further subdivide the mesh into submeshes of size $\sqrt{n} \times \sqrt{n}$ each.

In this context, it is important for every processor to determine which submesh it belongs to, as well as its local coordinates within that submesh. The problem can be abstracted as follows: for every j $(0 \le j \le n-1)$, compute $\left\lfloor \frac{j}{\sqrt{n}} \right\rfloor$ and $j \bmod \sqrt{n}$. Consider column j of an $n \times n$ mesh: every processor $P(i,j)$ $(0 \le i \le n-1)$ in that column computes $i \times \sqrt{n}$ and compares $i \times \sqrt{n}$ with j; clearly, $\left\lfloor \frac{j}{\sqrt{n}} \right\rfloor$ is the largest i for which $i \times \sqrt{n} \le j$ and, therefore, $j \bmod \sqrt{n} = j - \left\lfloor \frac{j}{\sqrt{n}} \right\rfloor \times \sqrt{n}$. To summarize the above discussion we state the following result.

Lemma 1. For every j $(0 \le j \le n-1)$, $\left\lfloor \frac{j}{\sqrt{n}} \right\rfloor$ and $j \bmod \sqrt{n}$ can be obtained in O(1) time on a reconfigurable mesh of size $n \times n$. □

Next, let us restrict our attention to integers in the range from 0 to $n-1$. As it turns out, it is sometimes convenient to be able to represent such an integer in *radix* $\sqrt{n}$ form. That is, we want to decompose an integer a $(0 \le a \le n-1)$ into a unique ordered pair of integers $<x, y>$ such that $a = x \times \sqrt{n} + y$ with $0 \le x \le \sqrt{n} - 1$ and $0 \le y \le \sqrt{n} - 1$. Once such a representation is available, processing a amounts to processing x and y. This representation will occur frequently in this paper. Note that, in the above notation,

$$x = \left\lfloor \frac{a}{\sqrt{n}} \right\rfloor \text{ and } y = a \bmod \sqrt{n}.$$

By Lemma 1, the unique ordered pair $<x_j, y_j>$ of a_j $(0 \le j \le n-1)$ can be obtained in O(1) time on a reconfigurable mesh of size $n \times n$.

Recently, Olariu, Schwing, and Zhang [7] pointed out that on a reconfigurable mesh of size $m \times n$, the prefix sums of a binary sequence $b_0, b_1, \ldots, b_{n-1}$ with b_j stored by processor $P(0,j)$ $(0 \le j \le n-1)$ can be computed in O(log n) time if $m=1$ and in $O(\frac{\log n}{\log m})$ if $m>1$. An important corollary of this result goes as follows.

Lemma 2. The prefix sums of a binary sequence $b_0, b_1, \ldots, b_{n-1}$ with b_j stored by processor $P(0,j)$ $(0 \le j \le n-1)$ can be computed in O(1) time on a reconfigurable mesh of size $\sqrt{n} \times n$. □

We next present another important application of the binary prefix sums algorithm in [7]. Consider a reconfigurable mesh M of size $n \times m$ with $m \le n$, and an integer sequence $x_0, x_1, \ldots, x_{m-1}$ with $0 \le x_i \le \frac{n}{m} - 1$ for all values of i. We assume that the sequence is stored by the processors in the first row of M, with $P(0,i)$ storing x_i, for all i. Note that if m is small, that is, $\log m \le \frac{\log n}{\log m}$, we can compute the prefix sums by using the general prefix sums algorithm discussed in [5]. We therefore, assume that $\log m > \frac{\log n}{\log m}$.

To begin, establish vertical busses in all columns of M and let every processor $P(0,i)$ broadcast x_i southbound to $P(i\frac{n}{m} + x_i, i)$. Next, establish horizontal busses in all rows $i\frac{n}{m} + x_i$ of M and let processor $P(i\frac{n}{m} + x_i, i)$ broadcast x_i to $P(i\frac{n}{m} + x_i, 0)$.

---

[*] to avoid handling tedious housekeeping details we assume, without loss of generality, that n is a perfect square.

Next, every processor $P(j,0)$ writes a 1 or a 0 into a local variable depending on whether or not $i\frac{n}{m}<j\le i\frac{n}{m}+x_i$, for some i. What results, is a binary sequence in the first column of M. It is easy to confirm that, having computed the prefix sums of this binary sequence, the value of the prefix sum stored by every processor $P(i\frac{n}{m}+x_i,0)$ is precisely the value of the prefix sum $x_0+x_1+...+x_i$.

Now reversing the broadcasting operations detailed above, we store the corresponding prefix sums in all the processors $P(0,i)$. By Lemma 2, the entire computation above takes $O(\frac{\log n}{\log m})$ time. Consequently, we have the following result.

Theorem 3. On a reconfigurable mesh of size $n\times m$, the prefix sums of an m-element integer sequence $x_0, x_1, ..., x_{m-1}$ with $0\le x_i\le\frac{n}{m}-1$ can be solved in $O(\min\{\log m, \frac{\log n}{\log m}\})$ time. □

Theorem 3 implies the following corollary that will be used again and again.

Corollary 3.1. The prefix sums of an integer sequence $x_0, x_1, \ldots, x_{\sqrt{n}-1}$ with $0\le x_i\le\sqrt{n}-1$ can be computed in O(1) time on a reconfigurable mesh of size $\sqrt{n}\times n$. □

3. Prefix sums of an integer sequence

Given a sequence $a_0, a_1, \ldots, a_{n-1}$ of integers with $0\le a_j\le n-1$, our task is to compute all the prefix sums a_0, a_0+a_1, $a_0+a_1+a_2$, ..., $a_0+\cdots+a_{n-1}$. In this section, we propose an algorithm to solve the prefix sums problem stated above in O(1) time on a reconfigurable mesh R of size $n\times n$. For this purpose it is assumed that the sequence is stored in the first row of R, with a_i being stored by processor $P(0,i)$ for all i. At the end of the computation, every processor $P(0,i)$ will store $a_0+\cdots+a_i$.

For every a_i, the processors in column i compute the tuple $<x_i,y_i>$ such that $a_i=x_i\times\sqrt{n}+y_i$, with $0\le x_i\le\sqrt{n}-1$ and $0\le y_i\le\sqrt{n}-1$; by Lemma 1, this can be done in O(1) time on R. It is easy to see that the problem of computing the prefix sums of the original sequence is reduced to the problem of computing the prefix sums of the sequences $x_0, x_1, \ldots, x_{n-1}$ and $y_0, y_1, \ldots, y_{n-1}$. We only demonstrate how to compute the prefix sums $x_0+x_1+\cdots+x_i$ $(0\le i\le n-1)$, for the prefix sums of $y_0, y_1, \ldots, y_{n-1}$ can be computed in the same way.

To begin, partition the $n\times n$ mesh R into $\sqrt{n}$ submeshes $M_0, M_1, \ldots, M_{\sqrt{n}-1}$ of size $n\times\sqrt{n}$, with M_i involving the columns $i\sqrt{n}$ through $(i+1)\sqrt{n}-1$ of R, and compute the prefix sums of the elements stored in the first row of every M_i. Now Corollary 3.1 guarantees that this can be done in O(1) time in every submesh M_i. Let $c_0, c_1, ...c_{\sqrt{n}-1}$ be a new sequence of integers with c_i $(0\le i\le\sqrt{n}-1)$ standing for $x_{i\sqrt{n}}+x_{i\sqrt{n}+1}+\cdots+x_{(i+1)\sqrt{n}-1}$. Put differently, every c_i is the sum of the elements in the first row of M_i.

The purpose of the next data movement operation is to move all the c_i's to the first row of M_0, with $P(0,i)$ storing c_i for all i. We proceed as follows: every processor $P(0,(i+1)\sqrt{n}-1)$ holding c_i, broadcasts c_i to $P(i,(i+1)\sqrt{n}-1)$; in turn, $P(i,(i+1)\sqrt{n}-1)$ broadcasts c_i to processor $P(i,i)$; finally, $P(i,i)$ broadcasts c_i to $P(0,i)$.

It is important to note that every c_i is an integer in the range 0 to $n-1$. As before, every processor $P(0,i)$ $(0\le i\le\sqrt{n}-1)$ computes the tuple $<u_i,v_i>$ such that $c_i=u_i\times\sqrt{n}+v_i$, with $0\le u_i\le\sqrt{n}-1$ and $0\le v_i<\sqrt{n}-1$; by Lemma 1, this can be done in O(1) time on M_0. Consequently, computing the prefix sums of $c_0, c_1, \ldots, c_{\sqrt{n}-1}$ reduces to computing the prefix sums of $u_0, u_1, \ldots, u_{\sqrt{n}-1}$ and $v_0, v_1, \ldots, v_{\sqrt{n}-1}$. By Corollary 3.1, this operation, performed in two stages, takes O(1) time altogether.

Now performing in reverse the data movement operation detailed above, we can arrange for $c_0+c_1+...+c_i$ $(0\le i\le\sqrt{n}-1)$ to be moved to $P(0,(i+1)\sqrt{n}-1)$. All that remains to be done is to add, for all i $(0\le i\le\sqrt{n}-2)$, the value just received by $P(0,(i+1)\sqrt{n}-1)$ to all the processors in the first row of M_{i+1}. This, of course, is done in a straightforward way in O(1) time.

To summarize our findings we state the following result.

Theorem 4. The prefix sums of a sequence of integers $a_0, a_1, \ldots, a_{n-1}$ with $0\le a_i\le n-1$ can be computed in O(1) time on a reconfigurable mesh of size $n\times n$. □

It is a simple observation that Theorem 4 can be extended to handle integers in the range 0 through n^c by handling every digit in the radix $\sqrt{n}$ representation separately. This, obviously, takes

O(c) time on a reconfigurable mesh of size $n \times n$.

4. Packing

Let $X = \{x_0, x_1, ..., x_{n-1}\}$ be an arbitrary n-element set and let $P_0, P_1, ..., P_{m-1}$ ($m \leq n$) be an arbitrary collection of predicates[*] such that every element of X satisfies at most one such predicate. An element of X that satisfies a predicate is referred to as *regular*; elements of X satisfying no predicate are termed *special*. We tacitly assume that testing whether an input element satisfies a predicate takes O(1) time. In this terminology, the packing problem asks for a partition of X such that regular elements satisfying the same predicate occur in the same subset of the partition. Similarly, the special elements occur in a single subset of the partition.

In this section we show that the ability to compute the prefix sums of an integer sequence affords us an efficient packing algorithm. For this purpose, consider a reconfigurable mesh R of size $n \times n$, with X stored by the processors in the first row of R, with $P(0,i)$ storing x_i for all i.

Our plan is to set up m buckets, one for each predicate in the collection, and to distribute the regular elements into the corresponding buckets. An additional bucket will be set up for the special elements. Once this is done, we arrange the buckets in the first row of R in some order. We begin the algorithm by grouping input elements into intermediate buckets. The idea is to lump together predicates in batches of size $\sqrt{n}$. For later reference we note that for all i ($0 \leq i \leq \sqrt{n} - 1$), the i-th batch contains the predicates $P_{i\sqrt{n}}, P_{i\sqrt{n}+1}, ..., P_{(i+1)\sqrt{n}-1}$. The elements of X are then placed into the intermediate buckets so that each of the elements of the i-th bucket satisfies one of the predicates in the i-th batch of predicates. In the second stage of our algorithm, the intermediate buckets will be further refined to obtain the desired result.

We assume that the m predicates have been broadcast to the first m rows of R such that all processors in row i ($0 \leq i \leq m-1$) know predicate P_i. To make the presentation easier to follow we assume, as a technicality, that the rows m through $n-1$ have been distributed a dummy predicate that is always false.

We let every processor $P(0,j)$ ($0 \leq j \leq n-1$) broadcast x_j to the whole column j. Now every processor $P(i,j)$ marks itself whenever x_j satisfies predicate P_i. Observe that by our assumption, at most one processor is marked in every column j.

At this moment, it is convenient to view the mesh R as consisting of submeshes $R_0, R_1, ..., R_{\sqrt{n}}-1$ of size $\sqrt{n} \times n$, with R_i ($0 \leq i \leq \sqrt{n} - 1$) consisting of rows $i\sqrt{n}$ through $(i+1)\sqrt{n} - 1$ of R.

Next, in every submesh R_i vertical subbusses are established and every marked processor in column j of in R_i broadcasts x_j northbound to $P(i\sqrt{n}, j)$; upon receipt of x_j, processor $P(i\sqrt{n}, j)$ marks itself. Every processor in row $i\sqrt{n}$ ($0 \leq i \leq \sqrt{n} - 1$) writes a 1 or a 0 into a local variable depending on whether or not it is marked. Now compute, in every R_i, the prefix sums of the resulting binary sequence in row $i\sqrt{n}$. By Lemma 2, this can be done in O(1) time. Note that the value of the prefix sum at every marked processor $P(i\sqrt{n}, j)$ specifies the unique position of x_j in the corresponding intermediate bucket. We want all the intermediate buckets arranged in the first row of R. For this, we need the starting position of every such bucket. In turn, this is computed as follows.

For every i, let r_i be the value of the prefix sum at the rightmost marked processor in row $i\sqrt{n}$. Next, compute the prefix sums of the integer sequence $r_0, r_1, ..., r_{\sqrt{n}-1}$. By Theorem 4, this takes O(1) time on the mesh R.

It is important to note that for all i, r_i stands for the number of elements in the input sequence that satisfy one of the predicates $P_{i\sqrt{n}}, P_{i\sqrt{n}+1}, ..., P_{(i+1)\sqrt{n}-1}$, that is the i-th intermediate bucket. By the same token, the value of $r_0 + r_1 + ... + r_{i-1}$ is the position, in the first row of R, beginning at which the elements of the i-th intermediate bucket should be recorded. We now broadcast the value $r_0 + r_1 + ... + r_{i-1}$ to all the marked processors in row $i\sqrt{n}$, and let every such processor add this value with the value of the binary prefix sum it stores. For every marked processor this yields a unique position in the first row of R.

---

[*] note that the enumeration of sets does not imply that an order relation among the sets exists.

We are now ready to handle the special elements. Observe, that is an easy matter for every processor $P(0,i)$ to mark itself if x_i is special. Now let every processor in the first row of R set a local variable to 1 or 0 depending on whether or not the input element it holds is special or not. After having computed the prefix sums of the resulting binary sequence every special element knows its rank among the special elements. Furthermore, by adding $r_0+r_1+...+r_{\sqrt{n}-1}$ to this rank gives the exact position of every special element in the first row of R. The special elements need no further processing, as their intermediate bucket coincides with the final bucket. It is a straightforward data movement operation, whose details are omitted, to broadcast all the elements of the input sequence to their corresponding positions in the first row of R as described above.

Let $x'_0, x'_1, ..., x'_{n-1}$ be the resulting permutation of X, and let $p_0, p_1, ..., p_{\sqrt{n}-1}$ be the starting positions of the intermediate buckets determined in the previous computational step. Put differently, the elements x'_j with $p_0 \le j < p_1$ satisfy one of the first $\sqrt{n}$ predicates, elements x'_j with $p_1 \le j < p_2$ satisfy the second batch of $\sqrt{n}$ predicates, and so on.

To further refine the intermediate buckets it is convenient to consider the mesh R as consisting of submeshes $S_0, S_1, ..., S_{\sqrt{n}-1}$ with S_0 involving columns p_0 through p_1-1, S_1 involving columns p_1 through p_2-1, and so on. Note that the elements of the i-th intermediate bucket are in the first row of S_i. The i-th batch of predicates will be distributed to the rows of S_i as follows: for all j $(0 \le j \le \sqrt{n}-1)$, row $j\sqrt{n}$ of S_i receives the j-th predicate of the i-th batch.

Now repeat the computation detailed in the first part of the algorithm in every submesh S_i. It is easy to confirm that the outcome is a permutation $x''_0, x''_1, ..., x''_{n-1}$ of X such that elements satisfying the same predicate occur consecutively in this permutation; similarly, the elements that satisfy no predicate in the original collection (i.e. the special elements) occur consecutively, in some order, after all the regular elements.

To summarize our discussion we state the following result.

Theorem 5. Given an n-element set X and a collection of $m \le n$ predicates such that every element of X satisfies at most one predicate, the packing problem can be solved in O(1) time on a reconfigurable mesh of size $n \times n$. □

5. Applications

5.1. Integer sorting

We assume a reconfigurable mesh R of size $n \times n$. The input is an integer sequence $a_0, a_1, \ldots, a_{n-1}$ with $0 \le a_j \le n-1$, stored in the first row of R; it is further assumed that for all i, processor $P(0,i)$ stores a_i. Our task is to sort the input sequence in ascending order. This problem arises in many important applications in pattern analysis, image processing, graph theory, and others. Clearly, we could accomplish our task by using a general-purpose sorting algorithm; however, since the input is of a rather special nature one would expect to be able to obtain a more efficient solution. As we are about to explain, this is indeed the case. Specifically, we show that the solution to the packing problem affords us a constant time sorting algorithm on the original $n \times n$ mesh R.

Consider the set of predicates $P_0, P_1, ..., P_{n-1}$, with P_i $(0 \le i \le n-1)$, defined as

$$P_i \equiv x = i,$$

and solve the resulting packing problem.

Notice that in our solution to the packing problem the elements satisfying predicate P_i occur before the elements satisfying P_j whenever $i<j$. Consequently, the solution to the above packing problems is exactly the solution to the sorting problem. The above discussion can be summarized as the following theorem.

Theorem 6. A sequence of n integers in the range 0 through $n-1$ can be sorted in O(1) time on a reconfigurable mesh of size $n \times n$. □

Note that the range of the integers to be sorted can be extended to n^c for arbitrary c; in this case the complexity of our algorithm is O(c) on a reconfigurable mesh of size $n \times n$.

5.2. Representation conversion for graphs and trees

A most desirable representation of a directed graph is the well-known adjacency list representation [1]. However, as it turns out, this is not always the input format of the graph, particularly, when the graph is produced in an intermediate step of some computation. Typically, a directed graph is specified by giving a list of nodes along with a list of directed edges (i.e. ordered pairs of nodes of the graph). As we are about to point out, converting the list-of-edges representation to the adjacency list representation is a simple application of packing.

Let G be a directed graph with n nodes and m edges. The nodes are assumed to be $x_0, x_1, ..., x_{n-1}$; the directed edge from x_i to x_j is specified as an ordered pair (x_i, x_j). Consider the set of predicates $P_0, P_1, ..., P_{n-1}$, with P_i $(0 \leq i \leq n-1)$, defined as

$$P_i \equiv \text{the first component of edge } z \text{ is } x_i .$$

It is easy to confirm that after solving the corresponding packing problem on a reconfigurable mesh of size $\max\{n,m\} \times m$ we obtain the adjacency list representation of G. The above discussion can be summarized as follows.

Theorem 7. The edge-list representation of a directed graph with n nodes and m edges can be converted into an adjacency-list representation in O(1) time on a reconfigurable mesh of size $\max(m,n) \times m$. □

Rooted trees are typically represented by parent pointers; that is, every node in the tree stores a pointer to its parent. This primitive representation, however, is not very convenient in application where a more sophisticated representation is desirable. In *standard form* a rooted (ordered) tree is stored in an unordered array, with every node having a parent pointer along with a doubly linked list of children. The standard form is convenient for application of many techniques such as Euler-tour technique in designing parallel algorithms. The problem of converting from the parent-pointer representation to the standard form can be handled easily by packing the nodes in the tree by parent.

Specifically, consider a (rooted) tree T with nodes n $x_0, x_1, ..., x_{n-1}$ specified by a set of tuples, $<u, \text{parent}(u)>$. with $P(0,j)$ storing $<x_j, \text{parent}(x_j)>$.

The output is a representation of T in a standard form featuring, for every node u, in addition to parent(u), the first child f_child(u), its left sibling l_sibling(u), and its right sibling r_sibling(u).

To achieve this, consider the set of predicates $P_0, P_1, ..., P_{n-1}$, with P_i $(0 \leq i \leq n-1)$, defined as

$$P_i \equiv \text{the parent of } u \text{ is } x_i .$$

It is easy to confirm that after solving the corresponding packing problem on a reconfigurable mesh of size $n \times n$, nodes with the same parent pointer, (i.e. siblings in the tree) are stored in contiguous processors of the first row of the mesh. Therefore, the left sibling of u is stored in the processor immediately to its left (provided they have the same parent in the tree). The right sibling and the first child of u are is determined similarly.

The above discussion about the representation conversion of a tree can be summarized as follows.

Theorem 8. Converting from the parent-pointer representation of a rooted n-node tree to the standard form takes O(1) time on a reconfigurable mesh of size $n \times n$. □

5.3. The perimeter and the area of a binary image

One of the basic problems in image processing involves computing the area and the perimeter of a binary image. The area of a binary image is defined as the number of pixels contained within its boundary. The perimeter of an image is the length of its boundary, i.e. the number of pixels on the boundary. We assume an 0/1 image of size $n \times n$ with noise eliminated. The boundary of the image can be found by checking a constant number of neighbors of each pixel. This problem can be abstracted as follows. Given an array $A[0..n-1, 0..n-1]$ with $A[i,j]$ $(0 \leq i \leq n-1,\ 0 \leq j \leq n-1)$ being 0 or 1, compute the number of 1's in A. In this section, we show how to solve this problem in $O(\log\log n)$ time on a reconfigurable mesh of size $n \times n$ with $P(i,j)$ storing $A[i,j]$. The result will be stored by $P(0,0)$.

Our algorithm proceeds by recursion. We first divide the original reconfigurable mesh into $\sqrt{n}$ submeshes, $M_0, M_1, \ldots, M_{n-1}$, of size $\sqrt{n} \times n$, where M_k involves the columns from $k\sqrt{n}$ to $(k+1)\sqrt{n}-1$; next subdivide every M_k into $\sqrt{n}$ submeshes of size $\sqrt{n} \times \sqrt{n}$, $M_{k0}, M_{k1}, \ldots, M_{kn-1}$, where

M_{st} consists of the processors $P(i,j)$ for $s\sqrt{n} \le i \le (s+1)\sqrt{n}-1$ and $t\sqrt{n} \le j \le (t+1)\sqrt{n}-1$. We now compute the number of 1's in M_{st} recursively; let the result be a_{st}, which is stored by $P(s\sqrt{n},t\sqrt{n})$. Notice that the number of 1's in the original reconfigurable mesh is given by $\sum\limits_{0 \le s,t \le \sqrt{n}-1} a_{st}$. This yields the final step: send a_{st} $(0 \le s,t \le \sqrt{n}-1)$ from $P(s\sqrt{n},t\sqrt{n})$ to $P(0,s\sqrt{n}+t)$; compute $\sum\limits_{0 \le s,t \le \sqrt{n}-1} a_{st}$. Notice a_{st} in the range from 0 to n, $\sum\limits_{0 \le s,t \le \sqrt{n}-1} a_{st}$ can be computed in O(1) on the original reconfigurable mesh by Theorem 2.

With $T(n)$ denoting the complexity of computing the number of 1's in $A[0..n-1,0..n-1]$, we have the following recursion

$$T(n) = T(\sqrt{n}) + O(1).$$

Together with the basis $T(2) = O(1)$, we get $T(n) = O(\log\log n)$.

The following theorem captures our findings.

Theorem 9. The number of 1's in a 0/1 matrix of size $n \times n$ can be computed in O(loglog$n$) time on a reconfigurable mesh of size $n \times n$. □

Corollary 9.1. Both the area and the perimeter of a binary image of size $n \times n$ can be computed in O(loglog$n$) time on a reconfigurable mesh of size $n \times n$. □

6. Conclusions and open questions

To overcome the inefficiency of long distance communications among cooperating processors, bus systems have been recently added to a number of parallel machines. If such a bus system can be dynamically changed, under program control, to suit communication needs, it is referred to as *reconfigurable*.

In many important applications in pattern recognition, image processing, graph theory, and database design, the problem at hand can be reduced to a computational problem involving only integer values. These problems are collectively referred to as integer problems. As it turns out, it is often possible to design specialized algorithms to handle these integer problems more efficiently than would standard, general-purpose, algorithms.

In this paper, we presented a constant time algorithm to compute the prefix sums of an integer sequence on a reconfigurable mesh of size $n \times n$. We then showed that this first solution implies a constant time solution to the packing problem. In turn, this affords us a constant time integer sort algorithm as well as a number of constant time conversion algorithms for graphs and trees.

Finally, we demonstrated an O(loglog$n$) algorithm to compute the perimeter and the area of a binary image of size $n \times n$.

There are, however, a number of open problems that the present work does not address; first we showed that we can handle polynomial ranges for the integers: our algorithms handle integers in the range 0 through n^c for some c. At present, we do not know how to handle truly large integers. Another open problem has to do with the "dense" case: here we have n^2 pieces of information stored one per processor. How fast can sort be performed? Further, it is still an open problem whether computing the prefix sums of a sequence of n real numbers can be computed in constant time on a reconfigurable mesh of size $n \times n$.

References

1. S. G. Akl, *The design and analysis of parallel algorithms*, Prentice-Hall, 1989.

2. H. Li and M. Maresca, Polymorphic-torus network, *IEEE Transactions on Computers*, vol. C-38, no. 9, (1989) 1345-1351.

3. H. Li and M. Maresca, Polymorphic-torus architecture for computer vision, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 11, no. 3, (1989) 233-243.

4. M. Maresca and H. Li, Connection autonomy and SIMD computers: a VLSI

implementation, *Journal of Parallel and Distributed Computing*, vol. 7 (1989) 302-320.

5. R. Miller, V. K. Prasanna-Kumar, D. Reisis, and Q. F. Stout, Meshes with reconfigurable buses, *Proceedings of the International Conference on Parallel Processing*, vol. 1, (1988) 205-208.

6. R. Miller, V. K. Prasanna-Kumar, D. Reisis, and Q. F. Stout, Data movement operations and applications on Reconfigurable VLSI arrays, *Proceedings of the fifth MIT Conference on Advanced Research in VLSI*, (1988) 163-178.

7. S. Olariu, J. L. Schwing, and J. Zhang, Fundamental Algorithms on Reconfigurable Meshes, Technical Report, TR-091-034, Department of Computer Science, Old Dominion University, 1991.

8. J. Rothstein, Bus automata, brains, and mental models, *IEEE Trans. on Systems Man Cybernetics* 18, (1988).

9. Q. F. Stout, Meshes with Multiple Buses, *Proc. 27th IEEE Symp. on the Foundations of Comp. Science*, (1986) 264-273.

10. B. F. Wang, G.H. Chen, and F.C. Lin, Constant time sorting on a processing array with a reconfigurable bus system, *Information Processing Letters*, vol. 34, no. 4, pp. 187-192, (1990).

11. B. F. Wang, C. J. Lu, and G. H. Chen, Constant Time Algorithms for the Transitive Closure Problem and its Applications, *Proceedings of the International Conference on Parallel Processing*, 3, (1990) 52-59.

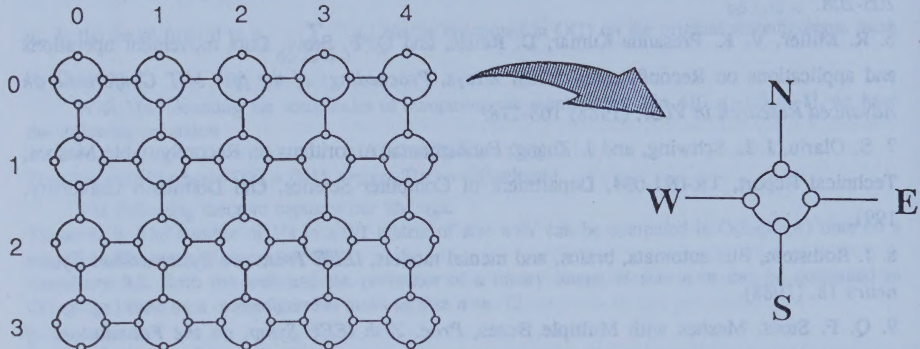

0
1
2
3
4
0
1
2
3
N
W
E
S

Figure 1. A reconfigurable mesh of size 4x5

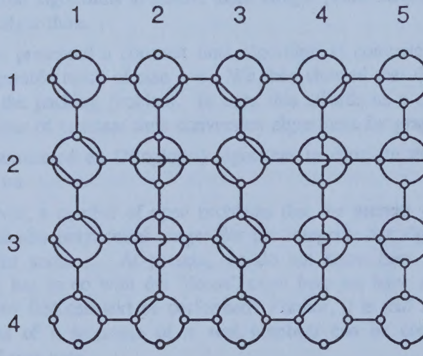


Figure 2. Examples of allowed connections

Parallel Circular-Arc Graph Algorithms on the Hypercube

M. G. Andrews
AT&T Bell Laboratories
PO Box 3033   2000 N. Naperville Rd.
Naperville, IL 60566-7033

D. T. Lee
Department of Electrical Engineering and Computer Science
Northwestern University   Evanston, IL 60208

Abstract

We consider the following problems when we are given a set of n circular-arcs: (1) the minimum cut sequence problem, (2) the minimum dominating set problem, (3) the minimum circle cover problem, (4) the maximum independent set problem, and (5) the minimum clique cover problem. We give a parallel algorithm of time complexity $O(\log^2 n)$ and processor complexity $O(n)$ for the minimum dominating set problem based on the hypercube model. For the minimum cut sequence, minimum circle cover, minimum clique cover problem, and maximum independent set problems, we give parallel algorithms of time complexity $O(\log n \log m)$ and processor complexity $O(n)$ with sorted input, where m is the size of the solution set.

Introduction

A graph $G = (V, E)$ is called a *circular-arc graph* if there is a one-to-one correspondence between the nodes of G and the members of a family $AF = \{A_0, A_1, \ldots, A_{n-1}\}$ of arcs on a circle such that two nodes i and j of G are adjacent if and only if the corresponding arcs A_i and A_j overlap. AF is called a circular arc model for the graph G. If no one arc contains another, AF is called a family of proper circular arcs and G is called a proper circular arc graph. See Figure A for an example of a circular-arc graph.

The *minimum cut sequence* (MCS) for circular arcs is defined as follows. Given a set, S, of n arcs $S = \{x_0, x_1, \ldots, x_{n-1}\}$ on a unit circle, find a set C of cut points $C = \{c_1, c_2, \ldots, c_m\}$ for some $m > 0$ such that each arc, X_i, contains at least one cut point and the cardinality of C is minimized. The number, m, is the minimum cut number of S and C is the minimum cut set of S. Lee, *et.al.*, give a sequential algorithm for finding the MCS in time $\theta(n \log n)$ [LSW]. Currently, there are no parallel algorithms for this problem.

A set, $S \subseteq V$, of vertices dominates a set $S' \subseteq V$ if every vertex in $S' - S$ is adjacent to some vertex in S. If S dominates V, we say that S is a dominating set for the graph G. A dominating set with minimum cardinality is called a *minimum dominating set* (MDS). Rau and Rangan

[RR] give a parallel algorithm for the MDS on circular-arc graphs which has time complexity $O(\log n)$ using $O(n/\log n)$ processors in the EREW-PRAM model.

A subset, S, of AF whose union equals the circle is called a *circle-cover* or a *cover*. A cover having the fewest arcs is called a *minimum circle cover* (MCC). The best sequential algorithm for MCC on circular-arc graphs has time complexity $O(n \log n)$ [LL]. For the EREW-PRAM model an algorithm of time $O(\log n)$ and processor complexity $O(n/\log n)$, with sorted input, was given by Atallah and Chen [AC].

A largest subset, S, of AF whose members are pairwise disjoint is called a *maximum independent set* (MIS). An algorithm for the EREW PRAM model was proposed by Rao and Rangan [RR]. They gave an optimal parallel algorithm of time complexity $O(\log n)$ and processor complexity $O(n/\log n)$, with sorted input.

A set of vertices, $S \subseteq V$, form a *clique* in G if every pair of vertices in S are adjacent. A *clique cover* of G is a partition of vertex set V into $V_1, V_2, \ldots, V_k$ such that each V_i, $1 \le i \le k$, forms a clique in G, k is the size of the clique cover. The *minimum clique cover* (MCQC) problem is to find a minimum clique cover for a graph. For circular-arc graphs, a parallel algorithm which runs in $O(\log n)$ time on a $O(n/\log n)$ processor EREW PRAM, with sorted input, was given by Rao and Rangan [RR].

For the above problems, the general solution is to delete any unnecessary arcs, find an appropriate successor for each arc (or cut interval). Then we use the successor function to find greedy solutions. We select the optimal solution from the greedy set. Most of the parallel algorithms given above use list ranking in combination with Euler tours to find the greedy solutions. List ranking has time complexity of $O(\log^3 n)$ on the hypercube [RJ]. For these five problems, a direct emulation of the PRAM algorithm on the hypercube would thus result in $O(\log^3 n)$ time complexity. However, by reducing the search space and using the monotonicity properties of the successor functions, we can reduce the time complexity by a logarithmic factor.

For the problems discussed here, we assume that the circular arc representation, AF, of the graph, G, is given. We assume an arbitrary origin, O, on the circle and each arc, A_i, will be denoted $[a_i, b_i]$, where a_i is its clockwise beginning point and b_i is its clockwise ending point. We let AF' denote the circular-arc graph AF with all containing arcs deleted. We let AF'' denote the circular-arc graph AF with all contained arcs deleted. We note that for proper circular-arc graphs, if the starting points of a set of arcs, $\{A_{i_1}, A_{i_2}, \ldots, A_{i_j}\}$, are ordered as $\{a_{i_1}, a_{i_2}, \ldots, a_{i_j}\}$ then the ending points of the arcs are also ordered in the same fashion. Without loss of generality, we assume that no single arc, A_i, covers the entire circle. We also assume that no two input arcs have the same endpoint.

Basic Hypercube Notation and Data Communication

A d-dimensional hypercube computer consists of $n = 2^d$ separate processing elements (or nodes), linked together in a d-dimensional binary cube network. Each node has associated with it a constant size memory. Basically, a 0-dimensional hypercube is just a single point, a 1-dimensional hypercube is a segment, a 2-dimensional hypercube is a square, a 3-dimensional hypercube is a regular cube, and, in general, a d-dimensional hypercube, C, consists of two *(d-1)*-dimensional hypercubes A and B with the corresponding vertices of each linked together. Each node in C is given a unique d-bit identification number (henceforth referred to as the node i.d.). Two nodes in a hypercube are said to be adjacent or neighboring if they share a link. In other words, two nodes are neighbors if and only if their corresponding node i.d.s differ in exactly one bit position. The communication diameter of hypercube networks is logarithmic.

A summary of the computational complexity of some basic hypercube primitives follows. Details can be found in [NS]. The following operations have time complexity $O(\log n)$ on an n-processor hypercube: Broadcasting, Concentration, Distribution, Generalization, MAX, MIN, Merging, Prefix, Postfix, Translation and Ranking. The following operations have time complexity $O(\log^2 n)$ on an n-processor hypercube: Sort, Random Access Read (RAR), and Random Access Write (RAW).

Minimum Cut Sequence

A cut interval is a nonempty angular interval, $c = (a, b)$, such that $a \in \{a_i \mid i = 0, 1, \ldots, n-1\} \cup \{b_i \mid i = 0, 1, \ldots, n-1\}$ and $\{b_i \mid i = 0, 1, \ldots, n-1\}$. In general, each b_i (or arc A_i) determines a cut interval. An arc, A_i, is covered by a cut interval, c_j, if any cut point within c_j intersects A_i.

As in [LSW], we define a successor function, as follows: The successor of a cut interval, $SUCC(c_i)$, is defined as: $SUCC(c_i) = \emptyset$ if there exists no arc not covered by c_i. Otherwise, $SUCC(c_i) = c_j$ where c_j is the first cut interval encountered when traversing the circle clockwise from c_i such that its CW end is determined by an arc not covered by c_i. The successor interval function on the cut intervals has the following monotonicity property. Consider any two cut intervals c_i and c_j. Let $c_k = SUCC(c_i)$ and $c_l = SUCC(c_j)$. If $c_i \leftrightarrow c_j \leftrightarrow c_k$, then either $c_l = c_k$ or $c_k \leftrightarrow c_l \leftrightarrow c_i$. ($\leftrightarrow$ denotes relative positioning on the circle).

For each c_i in AF', we define a minimal covering sequence as follows: $MCS(c_i) = (c_i, SUCC^1(c_i), \ldots, SUCC^j(c_i)$ such that j is the smallest integer so that all arcs are covered. Suppose that we compute $MCS(c_i)$ for all cut intervals, $0 \leq i \leq n-1$. Then, if we select the minimum sequence, we get a solution to the minimum cut problem. We will show later that only a subset of all $MCS(c_i)$ need to be computed.

Algorithm Min-Cut-Seq

```
begin:
    1. Delete containing arcs
    2. Determine the cut intervals
    3. Compute $SUCC(i)$ for $0 \leq i \leq n-1$
    4. Compute $|CS(i)|$ for $0 \leq i \leq SUCC(0)$
    5. Find a cut sequence with minimum cardinality
end.
```

First of all, the arcs which contain others can be deleted without affecting the existence or size of the minimum cut. This can be done in $O(\log n)$ time using the method of [RR]. After deleting the containing arcs, we relabel the surviving arcs so that their indices are in consecutive order $(A_0, A_1, \ldots)$. For simplicity of notation, we still use n to denote the number of arcs in AF'. Figure A shows the initial graph AF and Figure B shows the graph AF'.

To identify the cut intervals, we use the sorted endpoints of the arcs, PL'. Each cut interval is identified on the clockwise (cw) end by a right endpoint and on the counter clockwise (ccw) end by the endpoint directly preceding it. Parallel prefix is used to identify the endpoints in $O(\log n)$ time. Denote the arc index which corresponds to the right endpoint defining the cut, c_i, as $END(i)$. Next we use the fact that the starting point of a set of arcs are ordered in the same order as the ending points to derive the range of arcs "covered" by each cut interval. We use a parallel prefix to get the largest a_j value to the left of each b_i. Then the range of arcs covered is $END(i) \ldots j$. Note that if $END(i) > j$, then the set of arcs crosses the origin. Denote this range of arcs as $RANGE(i)$ where i is the index of the cut interval. $END(i)$ is the most counter clockwise (ccw) arc and j is the most clockwise (cw) arc. This step takes $O(\log n)$ time. Figure C shows the cut intervals for the graph AF'.

c_i	0	1	2	3	4	5	6	7	8
RANGE(i)(ccw,cw)	(7,0)	(8,0)	(0,1)	(1,3)	(2,4)	(3,4)	(4,4)	(5,5)	(6,7)
END(i)	7	8	0	1	2	3	4	5	6

Figure 1: Cut Intervals

We define $LEFT(0)$ as the set of all arcs that intersect with A_0 at its left endpoint. $LEFT(0)$ can be obtained in $O(\log n)$ time by broadcasting A_0 to all arcs A_i and having them check for intersection.

To obtain $SUCC(i)$, we merge two lists. The first list is the clockwise values of $RANGE$ and the second is the values of END. For END, we circularly translate the values by $(n - |LEFT(0)|)$ before merging. Then we use a parallel prefix to find the smallest END after each $RANGEcw$. The cut interval defined by END is then the successor of the cut interval associated with the $RANGEcw$. This takes $O(\log n)$ time. Thus, the cut intervals and their successors can be identified in $O(\log n)$ time given the sorted list of arcs.

c_i	0	1	2	3	4	5	6	7	8
$SUCC(c_i)$	3	3	4	6	7	7	7	8	1

Figure 2: Successors for Cut Intervals

From the definition of successor, we know: 1) a cut interval and its $SUCC$ do not intersect in the set of arcs they cover, and 2) the sequence $(c_i, SUCC(c_i))$ covers a contiguous range of arcs in AF'. Define k as the index of the most clockwise cut interval which covers A_0 ($k = SUCC(0) - 1$). Let m be the smallest cut interval index in $MCS(i) = (i, SUCC(i), \ldots, L)$. It is clear that $0 \leq m \leq k$. Using the definitions of successor and their monotonicity property, we can show that there is a minimum cut in $MCS(i), 0 \leq i \leq k$.

If m is the size of a minimum cut sequence, then, for all $j, 0 \leq j \leq n-1$, $(\mid MCS(c_j) \mid -m) \leq 1$, where $\mid MCS(c_j) \mid$ denotes the size of $MCS(c_j)$ which is as defined earlier. The proof follows from the definition of the successor function and the monotonicity property.

To compute $|MCS(i)|$ for $0 \leq i \leq k$, construct a successor forest, $FCS(V, E)$, from the $SUCC$ relations. $V = \{0, 1, \ldots, n-1\}$. For $0 \leq i \leq n-1$, if $i < SUCC(i)$, then $(i, SUCC(i))$ is an edge of E (Figure D). Define $R(i)$ to be the root of the tree to which vertex i belongs. Then, for $0 \leq i \leq k$ there is a cut sequence in $i \rightarrow SUCC(i) \rightarrow \ldots \rightarrow R(i)$ and $i \rightarrow SUCC(i) \rightarrow \ldots \rightarrow R(i) \rightarrow SUCC(R(i))$. Thus, for $0 \leq i \leq k$, $|MCS(c_i)|$ is obtained from the level of c_i in the forest. To compute the level of each node in FCS, we apply the following tree ranking algorithm. As input to the algorithm, we have the cut intervals of AF' in sorted order. For each of these cut intervals, if $i < SUCC(i)$, then we set

$PTR(i) = SUCC(i)$, else set $PTR(i) = \infty$. Because of the monotonicity property, we have the cut intervals also sorted in order of increasing value of $PTR(i)$.

Algorithm TreeRank

Input: A set of nodes sorted in increasing order of the value of $PTR(i)$, their ancestor.

Output: For each node, its LVL in the forest and its $ROOT$ node.

begin:

1. Identify the k unique values of PTR (the ancestors) and concentrate them in the first k PEs along with their PTR information and source PE index (to be used later in generalize step).
2. Using a distribute and then a concentrate operation, get the next level ancestor information.
3. Use a generalize operation to send the data for each value of PTR to the appropriate PEs.
4. Each PE which receives the data updates its ancestor data to be the ancestor of its ancestor, updates its level, merges the range of arcs covered, and updates its root information (if it has reached its root).
5. Repeat steps 1-4 until all nodes have determined their roots.

end.

Each of steps 1-4 take $O(\log n)$ time as each of the individual operations are either $O(1)$ or $O(\log n)$. Steps 1-4 are repeated $\log m$ times, where m is the height of the tree. The total time complexity is $O(\log m \log n)$. Thus, we can compute $LVL(i)$ and $ROOT(i)$ for all nodes in the forest in time $O(\log m \log n)$ with $O(n)$ processors, where m is the length of the longest path in the forest.

For $0 \leq i < SUCC(0)$, in constant time, we check if the sequence $(i, \ldots, R(i))$ covers all arcs. If so, $|MCS(i)| = LVL(i)$, otherwise, $|MCS(i)| = LVL(i) + 1$. We know the height of the FCS is no more than $|MCS| + 1$, therefore LVL and $ROOT$ for each node in the forest can be computed in time $O(\log m \log n)$, where $m = |MCS|$. Further, we can compute $|MCS(i)|$ for $0 \leq i \leq k$, in time $O(\log m \log n)$ with $O(n)$ processors, where m is the size of the minimum cut sequence.

To get the minimum cut sequence, select MIN($|MCS(i)|, 0 \leq i \leq k$) in $O(\log n)$ time and then mark the elements in the sequence using the following algorithm.

Algorithm Cut-Sequence

Input: The set of nodes, sorted by ancestor, and the index of the node which corresponds to the cut sequence, mincvr. For each node, the data $ROOT(i)$ is available.

Output: The arcs in the minimum cut sequence will be available in the first m, $m = |MCS(mincvr)|$ processors.

Descr: This algorithm is very similar to TreeRank except that: 1) we begin the $PATH$ in the node whose index is $mincvr$, 2) further nodes in the $PATH$ are marked as part of the distribute operation in step 2, and 3) we add a final step to concentrate all marked nodes in the first $|MCS(i)|$ processors.

Thus, the arcs of a cut sequence with minimum cardinality for a set of n circular-arcs can be found in time $O(\log m \log n)$ using $O(n)$ processors on a hypercube, where m is the size of the cut sequence.

Theorem 1 *The minimum cut sequence problem for a circular-arc graph can be solved in* $O(\log m \log n)$ *time with* $O(n)$ *processors with sorted input on the hypercube model where* m *is the size of the minimum cut sequence.*

Minimum Dominating Set

As in [RDL], we define the following three successor functions. For each arc in AF define $N(i)$ to be the arc, A_j, which is the first to end among all arcs which start after b_i. For each arc in AF define $F(i)$ to be the arc, A_j, to be the last arc (with respect to ending point) to extend A_i. For each arc in AF'', let $SUCC(i) = F(N(i))$. Also, $SUCC^1(i) = SUCC(i)$ and $SUCC^r(i) = SUCC(j)$ where $A_j = SUCC^{r-1}(i)$ for $r \geq 2$. SUCC has the following monotonicity property: Let $A_{i_1}, A_{i_2}, \ldots, A_{i_s} \in AF''$ such that $A_{i_1} \leq A_{i_2} \leq \ldots \leq A_{i_s}$. Then $SUCC(i_1)^k \leq SUCC(i_2)^k \leq \ldots \leq SUCC(i_s)^k$, for each $k = 1, 2, \ldots$.

For each A_i in AF'', we define the greedy dominating set $GD(i)$ as follows: $GD(i) = \{A_i, SUCC^1(i), \ldots, SUCC^r(i)\}$, where r is the smallest integer such that $GD(i)$ is a dominating set for AF. Define $RIGHT(0) = \{A_i | A_i \in AF''$ and A_i contains $b_0, i \neq 0\}$ and $LEFT(0) = \{A_i | A_i \in AF''$ and A_i contains $a_0, i \neq 0\}$. Let $MAIN(0) = \{A_0 \cup RIGHT(0) \cup LEFT(0)\}$. Then, a set of minimum size among $\{GD(i) | A_i \in MAIN(0)\}$ is an MDS for AF [RR].

The steps to this algorithm are essentially the same as for MCS except that we deal only with arcs, not cut intervals. First of all, the arcs which are contained within others can be deleted without affecting the existence or size of the minimum dominating set. We use the sorted list of endpoints of AF, PL. Rau and Rangan [RR] show that that the contained arcs can be identified by a algorithm which involves ranking, translation, and parallel prefix. All of these operations take $O(\log n)$ time on the hypercube.

To compute $SUCC(i)$ for all $A_i \in AF''$, we first compute $N(i)$ using the list, PL. We identify the containing arcs and determine the smallest index of all arcs contained within that arc. This can be done in $O(\log n)$ time by modifying the algorithm for MCS. Assign a weight to each left endpoint equal to its index and no weight to the right endpoints. If this is a containing arc, then assign a weight equal to the index of the smallest contained arc. Compute, using parallel postfix, for each right endpoint the smallest weight, w_i, to the right. Then $N(i) = w_i$. This operation takes $O(\log n)$ time.

To compute $F(i)$, we use the list PL and the knowledge that $F(i)$ is an arc in AF''. We assign a weight to each left endpoint of an arc in AF'' equal to its index. All right endpoints and left endpoints of contained arcs have a weight of zero. For each right endpoint, b_i, using parallel prefix, find the largest weight, w_i, that occurs before it in the list. Then $F(i) = w_i$. This operation takes $O(\log n)$ time on an n-processor hypercube. Finally, to compute $SUCC(i) = F(N(i))$, for all $A_i \in AF''$, we do a random access read (RAR). Each arc, A_i, goes to the processor according to $N(i)$ and gets the value of $F(N(i))$. This takes $O(\log^2 n)$ time on an n-processor hypercube and therefore, the total time to compute successors is $O(\log^2 n)$.

We delete the contained arcs and relabel the surviving arcs so that their indices are in consecutive order $(A_0, A_1, \ldots)$. For simplicity of notation, we still use n to denote the number of arcs in AF''.

To compute $|GD(i)|$, we first compute $|GD(i)|$ for all $A_i \in \{RIGHT(0), A_0\} = RT'(0)$. Then, we compute $|GD(i)|$, for all $A_i \in LEFT(0)$. To compute $|GD(i)|$ for $RT'(0)$, we construct a forest, FDS, as was done for FCS. It is easy to see that this forest has the monotonicity property. As for MCS, we can show that the size of a greedy solution differs by no more than 1 from the MDS, and thus, the height of the forest is $O(|MDS|)$. We can also show that for $i \in RT'(0)$, greedy solution in sequences $i \rightarrow SUCC(i) \rightarrow \ldots \rightarrow R(i)$ and $i \rightarrow SUCC(i) \rightarrow \ldots \rightarrow R(i) \rightarrow SUCC(R(i))$. To determine the arcs in the minimum sequences, we use an algorithm similar to MCS. Thus, the minimum greedy solution for $RT'(0)$ can be found in $O(\log m \log n)$ time where m is $O(|MDS|)$.

Next, we compute $|GD(i)|$ for $LEFT(0)$. We construct a forest, FDS, from the SUCC relations as follows: 1) $V = \{0, 1, \ldots, n-1\}$, 2) For $0 \leq i \leq n-1$, if ($i < SUCC(i)$ and $SUCC(i) \neq LEFT(0)$), then $(i, SUCC(i))$ is an edge of E. Also for $i \in LEFT(0)$, then $(i, SUCC(i))$ is an edge of E. Before running TreeRank algorithm, we translate the nodes of AF'' by $|LEFT(0)|$, this will cause the nodes in the tree to be sorted by increasing value of ancestor. We find the minimum greedy solution for $LEFT(0)$ as we did for $RT'(0)$ in $O(\log m \log n)$ time where m is $O(|MDS|)$. Then the MDS for AF is the smaller of GD(a) and GD(b).

Theorem 2 *The minimum dominating set problem for a circular arc graph can be solved in $O(log^2 n)$ time and $O(n)$ space using $O(n)$ processors in a hypercube model, whether or not the input is already sorted.*

Minimum Cover, Maximum Independent Set, and Minimum Clique Cover

The techniques used in [YCL] can be adapted to the hypercube model to give an algorithm of time complexity $O(\log n \log m)$ and processor complexity $O(n)$ for the MCC, MIS, and MCQC problems, where m is the size of the minimum cover. In all of these problems, we delete the appropriate set of arcs (either contained or containing), determine the successor relation, compute the size of n sequences of arcs, and finally, determine the set of arcs in the optimal sequence.

For MCC, we delete the contained arcs. For MIS and MCQC, the containing arcs are deleted. In the MCQC problem, for each deleted containing arc, A_i, there is an arc, A_j, in AF' such that A_j is properly contained in A_i. So, A_i is to be included in the clique to which A_j belongs.

A Successor relation for Minimum Cover, $SUCC$, is defined as follows. $SUCC(A_i) = A_j$ (or simply $SUCC(i) = j$) if A_i appears before A_j in clockwise direction and A_j reaches farthest and has nonempty intersection with A_i. Because of the ordering of the starting and ending points, for any two arcs, A_i and A_j, and their successors, $SUCC(A_i) = A_k$ and $SUCC(A_j) = A_l$, if $i < j$, then $k \leq l$. (Except for arcs which cross the origin.) All successors can be determined in $O(\log n)$ time using parallel prefix on the sorted list of appropriately weighted endpoints.

A *cover sequence*, $CS(i)$, for an arc, i, is defined as follows. Start from an arc A_i and then proceed in the clockwise direction, by selecting $SUCC(A_i)$ as the next arc in this cover, until the whole circle is covered. We compute $CS(i)$ for $0 \leq i \leq SUCC(0)$ and one of the resulting covers is the solution [YCL].

As in [YCL], to compute $|CS(i)|$ for $0 \leq i \leq SUCC(0)$, we first construct a successor forest, FMC as for MCS. Then, for $0 \leq i \leq SUCC(0)$, there is a cover in sequences

$i \rightarrow SUCC(i) \rightarrow \ldots \rightarrow R(i)$ and $i \rightarrow SUCC(i) \rightarrow \ldots \rightarrow R(i) \rightarrow SUCC(R(i))$. As input to the Tree-Rank algorithm, we have the arcs of AF'' in sorted order. Because of the monotonicity property, we have the arcs also sorted in order of ancestor. From [LL], we know that any greedy cover is no more than $|MCC| + 1$, thus, the length of the longest path in the tree is $O(|MCC|)$. Computing the arcs in the minimum cover is done similar to the MCS algorithm.

Theorem 3 *A minimum cover of a set of* n *arcs can be found in time* $O(\log n \log m)$ *using* $O(n)$ *processors on a hypercube, with sorted input, where* m *is the size of the minimum cover.*

A successor function for MIS is defined as follows: SUCC(A_i) is the arc, A_j, such that A_i and A_j are disjoint and A_j is "as near as possible" in the clockwise direction. For each arc, A_i, we determine the greedy MIS by arbitrarily selecting an arc, A_i, and then proceeding in the clockwise direction by selecting $SUCC(A_i)$ as the next arc until the newly containing arc cannot be added to this set (i.e., it intersects with some arc already in the sequence). [YCL] then show that it is necessary to consider only independent sequences with increasing arc indices.

To compute $SUCC(A_i)$, we use the sorted array of endpoints of AF', PL'. All successors can be determined by $O(\log n)$ time by a parallel postfix operation on the appropriately weighted list PL'.

To compute $|IIS(i)|$, the forest, FIS, is constructed as for MCS. Then, for any vertex i, the greedy MIS is given by the sequence $i \rightarrow SUCC(i) \rightarrow \ldots \rightarrow R(i)$ or the same sequence omitting $R(i)$. As input to the TreeRank algorithm, we use the nodes sorted by a_i, for which, therefore, ancestors are also sorted by a_i. Obviously, the the height of the tree m is $O(|MIS|)$. Computing the arcs in the maximum independent set is done similar to the MCS algorithm.

Theorem 4 *A maximum independent set for a set of* n *circular-arcs can be found in time* $O(\log n \log m)$ *using* $O(n)$ *processors on a hypercube, with sorted input, where* m *is the size of the MIS.*

The successor function for the MCQC problem is the same as that for MIS. A clique is defined linear if every arc in the clique contains a common point on the circle. For all $A_i \in AF'$, we define $LQ(i)$ to be the linear clique that is formed by all the arcs in AF' that contain b_i. For each A_i in AF', we define the sequence $GD(i) = \{A_{i_1}, A_{i_2}, \ldots, A_{i_k}\}$, where $A_{i_2} = SUCC^1(A_{i_1})$, etc. Then,the greedy clique cover is given by:

$$GD_q(i) = \begin{cases} \{LQ(i_1), LQ(i_2), \ldots, LQ(i_k)\}, & \text{if } SUCC(i_k) = A_i. \\ \{LQ(i_1), LQ(i_2), \ldots, LQ(i_k), LQ'(i_{k+1})\}, & \text{otherwise} \end{cases}$$

where $LQ'(i_{k+1})$=$LQ(i_{k+1})$-$LQ(i_1)$.

To form the greedy clique cover, we start with an arbitrary arc and then continue to select the successor of that arc until there is an intersection with the first arc. This sequence is $GD(i)$. Then we find the linear clique for each arc in the minimum sequence, $GD(a)$. It is not necessary to find all n greedy covers. We define LEFT(0) as the set of all arcs which contain a_0, the left endpoint of arc A_0. $LEFT' = \{A_0\} \cup LEFT(0)$. A clique cover of minimum size among $\{GD_q(i) | A_i \in LEFT'\}$ is an MCQC of AF' [RR].

After deleting the containing arcs, we renumber the arcs so that they are consecutive from 0 to $n-1$, where n is the number of vertices in AF'. For convenience, we continue to use n as the number of arcs.

To compute $|GD(i)|$, a forest, FCQC, is constructed from the SUCC relations as follows: 1) $V = (0, 1, \ldots, m-1)$, 2) For $0 \leq i \leq m-1$, if $i < SUCC(i)$ then $(i, SUCC(i))$ is an edge of E, except if $SUCC(i) \in LEFT(0)$, and 3) For $i \in LEFT(0)$, $(i, SUCC(i))$ is an edge of E. For the forest, we use the set of right endpoints in sorted order. With the right endpoints in this order, the ancestors will be in increasing order.

For $i \in LEFT'$, there is a clique cover in the sequences $i \rightarrow SUCC(i) \rightarrow \ldots \rightarrow R(i)$ and $i \rightarrow SUCC(i) \rightarrow \ldots \rightarrow R(i) \rightarrow SUCC(R(i))$. A algorithm similar to that for MCS is used to construct $GD(a)$, and the arcs in $GD(a)$ are so marked. Similar to MCS, we can show that the size of any greedy clique cover differs from $|MCQC|$ by no more than 1. Thus, the height of the tree is $O(|MCQC|)$.

To partition the arcs of AF among the linear cliques, we relabel the arcs according to their original labels. Then, in the sorted list of right endpoints assign a weight to each marked arc according to its position in the list. We use a parallel prefix to find the largest weight, w_i, that occurs before each right endpoint. If w_i=0, then assign a weight according to the weight of the last marked arc. All arcs with the same weight will be in a linear clique. This takes $O(\log n)$ time.

Theorem 5 *A minimum clique cover of a set of* n *circular-arcs can be found in* $O(\log n \log m)$ *time using* $O(n)$ *processors on a hypercube, where* m *is the size of the minimum clique cover and with sorted input.*

Summary

A set of circular-arc graph problems are solved in a unified manner. If the inputs are not sorted, the time complexity for all five problems is $O(\log^2 n)$. With sorted input, the time complexity is reduced to $O(\log n \log m)$ for the MCS, MCC, MIS, and MCQC problems, where m is the size of the solution set.

References

[AC] M. J. Atallah and D. Z. Chen, "An Optimal Parallel Algorithm for Minimum Circle-Cover Problem", *Information Processing Letters* 32 (1989) 159-165.

[LL] C. C. Lee and D. T. Lee, “On a Circle-Cover Minimization Problem,” *Information Processing Letters,* 18, No. 2, 1984, 1109-115.

[LSW] D. T. Lee, M. Sarrafzadeh, and Y. F. Wu, "Minimum Cuts for Circular-Arc Graphs", *SIAM J. Comuting*, 1990, Vol. 19, No. 6, 1041-1050.

[NS] D. Nassimi and S. Sahni, "Data Broadcasting in SIMD Computers", *IEEE Transactions on Computers*, Vol. C-30, No. 2, February, 1981, 101-107.

[RDL] C. Rhee, S. K. Dhall, and S. Lakshmivarahan, "Domination and Independence in Circular-Arc Graphs", unpublished manuscript, 1991.

[RR] A. Srinivasa Rao and C. Pandu Rangan, “Optimal Parallel Algorithms on Circular Arc Graphs,” *Information Processing Letters,* 33, 1989-90, 147-156.

[RJ] K. W. Ryu and J. JaJa, "Efficient Algorithms for List Ranking and for Solving Graph Problems on the Hypercube", *IEEE Transactions on Parallel and Distributed Systems*, Vol. 1, No. 1, January 1990.

[YCL] M. S. Yu, C. L. Chen and R. C. T. Lee, "Optimal Parallel Circle-Cover and Independent Set Algorithms for Circular Arc Graphs," *Proc. 1989 International Conference on Parallel Processing*, 1989, Vol. III, 126-129.

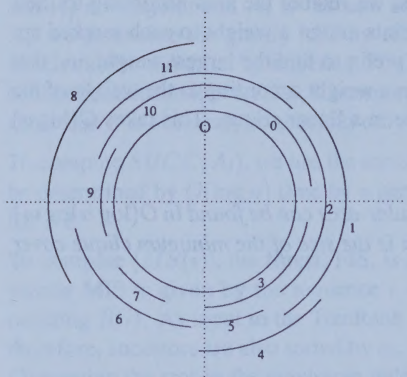

11
8
10
O
0
9
2
1
3
7
6
5
4

Figure A: A Circular-Arc Graph (AF)

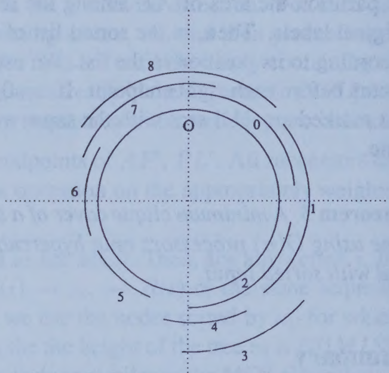

8
7
O
0
6
1
2
5
4
3

Figure B: Graph with Containing Arcs Deleted and Arcs Relabeled (AF')

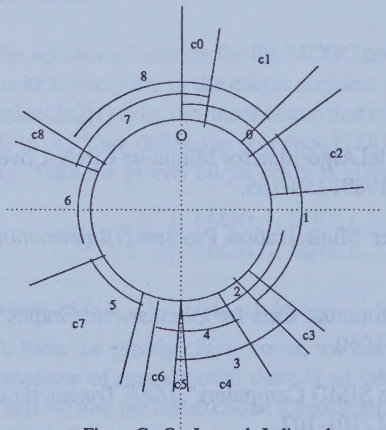

c0
c1
8
7
c8
O
0
c2
6
1
2
5
c7
4
c3
3
c6
c5
c4

Figure C: Cut Intervals Indicated

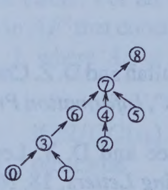


Figure D: FCS for Cut Intervals

A Hypercube Algorithm for the Maxima Problem

Si-Qing Zheng and *Fenglien Lee*

Department of Computer Science
Louisiana State University
Baton Rouge, LA 70803-4020

Let S be a set of N points in E^d. For a point $p \in S$, we use $x_k(p)$ to denote the k-dimensional coordinate value of p. For two distinct points p_i and p_j in S, we say that p_i *dominates* p_j if $x_k(p_i) \geq x_k(p_j)$, $k = 1, 2, \dots, d$. A point p in S is called a *maximum* of S if there does not exist any other point q of S such that q dominates p. The maxima problem consists of finding all the maxima of S. The best sequential algorithm is the one given in [4], which requires $O(N \log^{d-2} N + N \log N)$ time. In the last few years, there were several attempts on developing efficient parallel algorithms for the maxima problem ([1] [2] [3] [6] [7]). The only parallel algorithms on multiprocessors with sparse are the ones given in [3] and [7]. These two algorithms only suitable for the case $N = P$ and $d \leq 2$, where P is the number of processors. In this paper, we present an *SIMD* parallel hypercube implementation of the sequential algorithm given in [4]. We show that for a set of N points in E^d, our algorithm runs in $O(N^{\epsilon} \log^{d-1} N)$ time, where $\epsilon = \log_8 5$, assuming that there are N processors in the hypercube machine. For $N > P$, where P is the number of processors in the hypercube machine, our algorithm can be generalized to solve the maxima problem in $O((N/P^{1-\epsilon}) \log^{d-2} P \, (log(N/P) + \log P))$ time.

An n-dimensional hypercube machine Q_n consists of 2^n processors P_b, each has a unique n-bit binary address b, such that two processors are connected by a link edge if and only if their addresses differ in one bit position. Each processor is associated with a local memory. We define a k-dimensional subcube S_k of Q_n as a subnetwork containing processors (more precisely, a subgraph induced by vertices) such that the $(n-k)$-bit prefixes of their addresses equal to a given $(n-k)$-bit binary constant. A k-dimensional subcube containing processors $P_b, P_{b+1}, \dots, P_{b+2^{k-1}-1}$ is denoted as $S_k(b)$. We say that two k-dimensional subcubes $S_k(b_1)$ and $S_k(b_2)$ of Q_n are adjacent if there are two adjacent processors P_u and P_v such that P_u is in $S_k(b_1)$ and P_v is in $S_k(b_2)$. We use $(S_k(b_1), S_k(b_2))$ to denote the subnetwork of Q_n containing processors of $S_k(b_1)$ and $S_k(b_2)$. Clearly, the structure of $(S_k(b_1), S_k(b_2))$ is isomorphic to Q_{k+1}. We refer to $(S_k(b_1), S_k(b_2))$ as a *generalized subcube (generalized* $(k+1)$*–cube)* of Q_n.

The sequential maxima algorithm given in [4] consists of two levels of recursions, both having the divide-and-conquer feature. In the first level, the point set of a problem is partitioned into two subsets U, V of equal size such that all points in V have their x_d coordinate values no less than those of points in U. The sets M_U and M_V of maxima of U and V are computed independently. Then, M_U and M_V are "merged" into the the set M of maxima of the given problem. The second level of recursion is for the "merge" process. The function of this process is to find $M_V \cup \mathit{filter}_{d-1}[M_U \mid M_V]$, where $\mathit{filter}_{d-1}[M_U \mid M_V] = \{p \mid p \in M_U$ and p is not dominated in any dimension x_i, $1 \leq i \leq x_{d-1}$, by any point in $M_V\}$. This recursion is both on the size and the dimension of the problem. This algorithm is concisely described in [5]. The key issue in transforming this sequential algorithm into an efficient parallel one on the hypercube machine is to balance the computation work loads among processors and reduce the data communication as much as possible. By the recursive structure of the hypercube, it is desirable to partition it into subcubes of equal size, and allocate similar computation tasks to these subcubes. For the first level recursion, such a task/network partitioning is straightforward, since U and V are always respectively allocated to two $(m-1)$-cubes of a generalized subcube $(S_{m-1}(b_1), S_{m-1}(b_2))$. For the second level recursion, which corresponds to the "merge" step of the algorithm, we only need to consider how to parallelize $\mathit{filter}_{d-1}[S_{m-1}(b_1) \mid S_{m-1}(b_2)]$. We only discuss the case that $P = N$. Each processor's local memory initially contains one point. During the computation, several copies of a point may be created and distributed to other processors. We implement a stack in each local memory, each of its entries is used to keep a point. Hereafter, when we say the *current point* of processor P_i, we refer to the current top point entry of the stack associated with P_i. Subcube $S_{m-1}(b_1)$ (resp. $S_{m-1}(b_2)$) can be decomposed into four $(m$-3)-cubes $S_{m-3}(u_i)$ (resp. $S_{m-3}(v_i)$), where $u_i = b_1+(i-1)2^{m-3}$. (resp. $v_i = b_2+(i-1)2^{m-3}$), $i = 1, \dots, 4$. Each $S_{m-3}(u_i)$ (resp. $S_{m-3}(v_i)$) can be decomposed into two $(m$-4)-cubes $S_{m-4}(u_{i1})$ and $S_{m-4}(u_{i2})$ (resp. $S_{m-4}(v_{i1})$ and $S_{m-4}(v_{i2})$), where $u_{i1} = u_i$ and $u_{i2} = u_i+2^{m-4}$ (resp. $v_{i1} = u_i$ and $v_{i2} = v_i+2^{m-4}$). Clearly, $(S_{m-3}(u_i), S_{m-3}(v_i))$ is a generalized $(m-2)$-cube, and $(S_{m-4}(u_{ij}), S_{m-4}(v_{ij}))$ is a generalized $(m-3)$-cube. Let p^i denote the current point of processor P_{u_i}, $1 \leq i \leq 4$, and let $p^{\max}$ and q^{mid} denote the current points of $P_{b_1+2^{m-1}-1}$ and $P_{b_2+2^{m-2}}$, respectively. Comparing $x_k(q^{mid})$ with $x_k(p^i)$ and $x_k(p^{\max})$, exactly one of six following possibilities is true, assuming that all points are distinct in every coordinate. We identify the subproblems of $\mathit{filter}_k[S_{m-1}(b_1) \mid S_{m-1}(b_2)]$ for each case.

Case 1 : $x_k(q^{mid}) > x_k(p^{\max})$.
(1.1) $filter_{k-1}[S_{m-1}(u_1) \mid S_{m-2}(v_3)]$; and
(1.2) $filter_k[S_{m-1}(u_1) \mid S_{m-2}(v_1)]$.

Case 3 : $x_k(p^3) < x_k(q^{mid}) < x_k(p^4)$.
(3.1) $filter_k[S_{m-2}(u_3) \mid S_{m-2}(v_3)]$;
(3.2) $filter_{k-1}[S_{m-2}(u_1) \mid S_{m-2}(v_3)]$;
(3.3) $filter_k[S_{m-2}(u_1) \mid S_{m-2}(v_1)]$; and
(3.4) $filter_k[S_{m-3}(u_3) \mid S_{m-2}(v_1)]$.

Case 5 : $x_k(p^1) < x_k(q^{mid}) < x_k(p^2)$.
(5.1) $filter_k[S_{m-1}(u_1) \mid S_{m-2}(v_3)]$; and
(5.2) $filter_k[S_{m-3}(u_1) \mid S_{m-2}(v_1)]$.

Case 2 : $x_k(p^4) < x_k(q^{mid}) < x_k(p^{\max})$.
(2.1) $filter_k[S_{m-3}(u_4) \mid S_{m-2}(v_3)]$;
(2.2) $filter_{k-1}[\{S_{m-3}(u_1), S_{m-3}(u_2), S_{m-3}(u_3)\} \mid S_{m-2}(v_3)]$; and
(2.3) $filter_k[S_{m-1}(u_1) \mid S_{m-2}(v_1)]$.

Case 4 : $x_k(p^2) < x_k(q^{mid}) < x_k(p^3)$.
(4.1) $filter_k[S_{m-2}(u_3) \mid S_{m-2}(v_3)]$;
(4.2) $filter_k[S_{m-3}(u_2) \mid S_{m-2}(v_3)]$;
(4.3) $filter_{k-1}[S_{m-3}(u_1) \mid S_{m-2}(v_3)]$; and
(4.4) $filter_k[S_{m-3}(u_1) \mid S_{m-2}(v_1)]$.

Case 6 : $x_k(q^{mid}) < x_k(p^1)$.
(6.1) $filter_k[S_{m-1}(u_1) \mid S_{m-2}(v_3)]$.

For any case, we refer to each subproblem $filter_k$ (resp. $filter_{k-1}$) as a *task*. Each task can be further partitioned into a set of *subtasks*. Then, for each of these subtasks, eight subproblems $filter_k[S_{m-4}(u_{ij}) \mid S_{m-4}(v_{ij})]$ (resp. $filter_{k-1}[S_{m-4}(u_{ij}) \mid S_{m-4}(v_{ij})]$, $1 \le i \le 4$, and $j = 1, 2$, are created and solved in parallel, each on a generalized $(m-3)$-cube. Two pairs of subalgorithms *BROADCAST/CENSUS* and *SORT/UNSORT* are used for creating subtasks. Let $B(r/8)$ and $S(r/8)$, $r = 2^m$, denote the time required for a pair of *BROADCAST/CENSUS* and a pair of *SORT/UNSORT* (only needed when $filter_{k-1}[S_{m-4}(u_{ij}) \mid S_{m-4}(v_{ij})]$ are created), respectively. Then, the total time $F_k(r)$, $r = 2^m$, for $filter_k[S_{m-1}(b_1) \mid S_{m-1}(b_2)]$ is as follows.

$$F_k(r) = \begin{cases} 4F_k(r/8) + 4F_{k-1}(r/8) + 4S(r/8) + 8B(r/8), & \text{for } Case\ 1; \\ 5F_k(r/8) + 3F_{k-1}(r/8) + 3S(r/8) + 8B(r/8), & \text{for } Case\ 2; \\ 5F_k(r/8) + 2F_{k-1}(r/8) + 2S(r/8) + 7B(r/8), & \text{for } Case\ 3; \\ 5F_k(r/8) + F_{k-1}(r/8) + S(r/8) + 6B(r/8), & \text{for } Case\ 4; \\ 5F_k(r/8) + 5B(r/8), & \text{for } Case\ 5; \\ 4F_k(r/8) + 4B(r/8), & \text{for } Case\ 6. \end{cases}$$
$$\le 5F_k(r/8) + 4F_{k-1}(r/8) + 4S(r/8) + 8B(r/8).$$

Since $S(r) = O(log^2 r)$ and $B(r/8) = O(1)$, and $filter_2[S_{m-1}(b_1) \mid S_{m-1}(b_2)]$ takes $O(m^2)$ using the algorithm similar to that given in [7], we have $F_k(N) = O(5^{\log_8 N} \log^{k-1}(N))$. Let $T(N)$ be the total running time of our maxima algorithm, then $T(N) = T(N/2) + F_{d-1}(N) = O(N^{\in} \log^{d-1} N)$, where $\in = \log_8 5$. Since $F_k(r) \le 5F_k(r/8) + 4F_{k-1}(r/8) + 4S(r/8) + 8B(r/8)$, which determines the worst case performance of the algorithm, we can use this relation to enforce the algorithm operating in the *SIMD* fashion. The generalization of this algorithm for the case that $N > P$ is straightforward. Interested readers may refer to [8] for more details. The techniques used in our algorithms may be useful for constructing other efficinet hypercube algorithms. Note that the *CREW PRAM* maxima algorithm given in [1] runs in $O(log^{d-2} N)$ time. Using hypercube to simulate the *PRAM*, a polylogarithmic factor slowdown is possible. Such simulation results are based on packet-routing algorithms. It is not clear how to obtain an *SIMD* hypercube algorithm with a similar slowdown by simulating the existing *PRAM* maxima algorithms, if possible.

References

[1] M. J. Atallah, R. Cole, M. T. Goodrich, "Cascading Divide-and-Conquer: A Technique for Designing Parallel Algorithms", *Proc. of IEEE Symposium on Foundations of Computer Science*, October 1987

[2] M. J. Atallah and M. T. Goodrich, "Efficient Plane Sweeping in Parallel", *Proc. of ACM Symposium on Computational Geometry*, 1986

[3] F. Dehne, "O($n^{1/2}$) Algorithms for the Maximal Elements and ECDF Searching Problem on a Mesh-Connected Parallel Computer", *Information Processing Letters*, May 1986.

[4] H.T. Kung, F. Luccio and F. P. Preparata, "On Finding the Maxima of a Set of Vectors", *Journal of the ACM*, Oct. 1975.

[5] F. P. Preparata and M. I. Shamos, *Computational Geometry*, 2nd Edition, Springer-Verlag, 1988

[6] J. H. Reif and S. Sen, "Optimal Randomized Parallel Algorithms for Computational Geometry", *Proc. of Int'l Conf. on Parallel Processing*, 1987.

[7] I. Stojmenovic, "Computational Geometry on a Hypercube", *Proc. of Int'l Conf. on Parallel Processing*, 1988.

[8] S.-Q. Zheng and F. Lee, "A Hypercube Algorithm for the Maxima Problem", Technical Report #91-005, Dept. of Computer Science, LSU, 1991.

Simple and Efficient In-Place Parallel Sorting Algorithms

Yanjun Zhang
Department of Computer Science
Southern Methodist University
Dallas, TX 75275

Si-Qing Zheng
Department of Computer Science
Louisiana State University
Baton Rouge, LA 70803

An in-place parallel sorting algorithm is a sorting algorithm that requires $N + cp$ memory space, where N is the number of elements to be sorted, p is the number of processors used, and c is a small constant independent of N and p. In-place sorting algorithms are space-optimal. In a parallel computer system, there may be many processors and memory modules (internal memories). Ignoring the memory management issues by simply assuming that memories are inexpensive and the users can use as much internal memory space as they want may not be realistic for many applications. In fact, the space efficiency is more important and critical in parallel computing than that in sequential computing. An algorithm may perform poorly due to frequent I/O transfers caused by the limitation of internal memory space. Therefore, in-place parallel sorting may be desirable. Except previous work on *PRAM* models (e.g. [6]), very little is known about how to design efficient in-place parallel sorting algorithms on feasible parallel machines. In this paper, we present a simple and general method that can be used to construct efficient in-place parallel sorting algorithms.

Without loss of generality, we assume that t the number N of data elements to be sorted is a multiple of p. We only consider a particular type of parallel computer architectures, in which each processor PE_i is associated with a local memory module M_i. The processors are connected by an interconnection network. Each processor PE_i is assumed having a small constant number of working registers. Let $R_{i,1}$ and $R_{i,2}$ be two registers of PE_i explicitly used in our algorithm. We use $(R_{i,j})$ to refer to the content of $R_{i,j}$.

```
Algorithm INPLACE_SORT
    [Initialization]
    for all $PE_i$ do in parallel initialize$(M_i)$ endfor
    [Global Sorting]
    for $k = 1$ to $(N/p)\log_2 p$ do
        for all $PE_i$ do in parallel
            $R_{i,1} := deletmin(M_i)$; $R_{i,2} := deletmax(M_i)$
        endfor
        Using existing parallel sorting algorithm $A$ to sort numbers in all $R_{i,j}$'s in parallel so that
        $(R_{i,1}) < (R_{i,2})$ and $(R_{i,j}) < (R_{k,l})$ if $i < k$, where $1 \le i, k \le p$ and $1 \le j, l \le 2$,
        for all $PE_i$ do in parallel
            insert$((R_{i,1}), M_i)$; insert$((R_{i,2}), M_i)$;
        endfor
    endfor
    [Local Sorting]
    for all $PE_i$ do in parallel sort$(M_i)$ endfor
end INPLACE_SORT
```

This algorithm consists of two major steps. The first major step, which is referred to as *global sorting*, consists of $\log_2 p$ iterations. After this step is completed, any element in any M_i is no greater than any element in any M_j if $j \ge i$. In the second major step, which is referred to as *local sorting*, all PE_i sort elements in M_i in parallel, using procedure *sort*. Clearly, when this step terminates, all elements are sorted in non-decreasing order. The efficiency of algorithm *INPLACE_SORT* depends on the time required for initializing all M_i, global sorting and local sorting. We can maintain each M_i as a min-max heap [3]. A min-max heap is an implicit data structure which represents a set of elements with two partial orders using a linear array. Constructing a min-max heap of N/p elements in the initialization step can be done in $O(N/p)$ time in in-place fashion. Each of the operations $deletemin(M_i)$ [delete the minimum element from M_i], $deletemax(M_i)$ [delete the maximum element from M_i] and $insert((R_i), M_i)$ [insert (R_i) into heap M_i] can be done in $O(\log(N/p))$ time in in-place fashion. The local sorting can be carried out in $O((N/p)\log(N/p))$ time by running an optimal-time in-place sequential sorting algorithm *sort* in each processor. Therefore, if algorithm *INPLACE_SORT* is correct, then it runs in $O((N/p)\log p(\log(N/p)+f(p))+(N/p)\log(N/p))$ time, where $O(f(p))$ is the time of parallel algorithm A. Of course, algorithm A must be capable to sort $2p$ elements, two in each PE_i, in in-place fashion.

To prove the correctness of our algorithm, we resort to the 0-1 principle [7] to reduce the sorting problem with arbitrary inputs to the problem of sorting 0-1 inputs. For brevity, we omit this reduction, and state our main results derived from this reduction.

Theorem 1: Total number of $(N/p)\ \log_2 p$ iterations are sufficient for the global sorting of algorithm *INPLACE_SORT*.

Theorem 2: There exists problem instances which require at least $(N/(3p))\log_2 p$ global sorting iterations.

Theorem 3: For any sorting algorithm A on any parallel architecture with p processors that can sort $2p$ elements in $O(f(p))$ time and using O(1) space in each processor's local memory, the algorithm *IN_PLACE* constructed using A is an in-place algorithm which sorts N elements, $N > p$, on the same architecture in $\Theta(N/p)\log p\,(\log(N/p)+f(p))+(N/p)\log(N/p))$ time.

Our method gives rise to a set of space-optimal sorting algorithms on different parallel architectures. For example, consider the odd-even transportation sorting algorithm ([4, 7, 8]) on a linear array of p processors. This algorithm can be used as algorithm A in our transformation to obtain an in-place algorithm for sorting N elements, $N > p$, in $O((N/p)\log p\,(\log(N/p)+p) + (N/p)\log(n/p))$ time. If $p << (\log N)/p$, then the time complexity of this algorithm is $O((N/p)\log p \log(N/p))$; if $p >> (\log N)/p$, then the time complexity of this algorithm is $O(N\log p+(N/p)\log(N/p))$. Similarly, using the $O(p^{1/2})$-time parallel sorting algorithms given in [9] and [14] for a $p^{1/2} \times p^{1/2}$ mesh-connected multiprocessor system we can obtain in-place sorting algorithms that runs in $O((N/p^{1/2})\log p + (N/p)\log p\log(N/p))$ time. Consider hypercube machines. The bitonic sorting algorithm has running time $O((N/p)(\log(N/p)+\log^2 p))$, and it is not in-place. Using our method, we can convert the bitonic sorting algorithm into an in-place algorithm of time complexity $O((N/p)\log p\ (\log(N/p) + \log^2 p))$. Several other more efficient hypercube sorting algorithms, such as the ones given in [2, 5, 10, 12] can also be converted to in-place sorting algorithms using our method. Additional examples include the in-place sorting algorithms obtained by applying our method to known efficient sorting algorithms for multiprocessor structures as mesh-of-trees [11] and cube-connected-cycles [13]. For brevity, we omit detailed discussions. Currently, we are investigating a variant of our algorithm *IN_PLACE*. The new algorithm has all good features of *IN_PLACE*, and has several important applications. These results will be reported in a subsequent paper.

References

[1] S.G. Akl, *Parallel Sorting Algorithms*, Academic Press, New York.

[2] A. Aggarwal and M.-D. A. Huang, "Network Complexity of Sorting and Graph Problems and Simulating CRCW PRAMs by Interconnection Networks", in *Lecture Notes in Computer Science: VLSI Algorithms and Architectures (AWOC 88)*, J.H. Reif ed., Springer-Verlag, 339-350.

[3] M.D. Atkinson, J.R. Sack, N. Santoro, and T. Strothotte, "Min-Max Heaps and Generalized Priority-Queues, *Comm. ACM*, 29 (10).

[4] G. Baudet and D. Stevenson, "Optimal Sorting Algorithms for Parallel Computers", *IEEE Trans. Comput.*, C-(27 (1), 84-87.

[5] R.E. Cypher and J.L.C. Sanz, "Cobesort: An Optimal Sorting Algorithm for Feasible Parallel Computers", *Lecture Notes in Computer Science: VLSI Algorithms and Architectures (AWOC 88)*, J.H. Reif ed., Springer-Verlag, 456-464.

[6] X. Guan and M.A. Langston, "Time-Space Optimal Parallel Merging and Sorting", Technical Report CS-90-102, Dept. of Comput. Sci., Univ. of Tennessee, Knoxville.

[7] D.E. Knuth, *The Art of Computer Programming*, Vol.3, Addison-Wesley.

[8] H.T. Kung, "The Structure of Parallel Algorithms" in *Advances in Computers*, (M.C. Yovits, ed.), Academic Press, New York.

[9] D. Nassimi and S. Sahni "Bitonic Sort on a Mesh-Connected Parallel Computer", *IEEE Trans. Comput.*, C-28 (1) 2-7.

[10] D. Nassimi and S. Sahni "Parallel Permutation and Sorting Algorithms and a New Generalized Connection Network", *J. ACM*, 29 (3), 642-667.

[11] D.E. Muller and F.P. Preparata, "Bounds to Complexities for Sorting and for Switching" *J. ACM*, 22 (2), 195-201.

[12] C.G. Plaxton, "Efficient Computation on Sparse Interconnection Networks", Ph.D. thesis, Dept. of Comput. Sci., Stanford Univ., 1989.

[13] F.P. Preparata and J. Vuillemin, "The Cube-Connected Cycles: A Versatile Network for Parallel Computation", *Comm. ACM*, 24 (5), 300-309.

[14] C.D. Thompson and H.T. Kung "Sorting on a Mesh-Connected Parallel Computer", *Comm. ACM*, 20 (4), 263-271.

A Parallel Algorithm for the Class of $\pm 2^b$ ASCEND Computations on a SIMD Hypercube

David Nassimi
CIS Dept., New Jersey Institute of Technology
Newark, NJ 07102
Email: Nassimi@vienna.njit.edu

Abstract

We briefly describe an $O(n^2/\log n)$ algorithm for the class of $\pm 2^b$-ascend computations, assuming $O(\log n)$ space per PE, where $N = 2^n$ is the number of PEs. The SIMD hypercube model restricts all communications to a single dimension at a time. We refer the reader to [3] for details.

Summary

There is a wide class of parallel algorithms which require communications in the form of a 2^b permutation and -2^b permutation. Let $N = 2^n$ be the number of PEs, with each PE[i] containing a data element $A[i]$, $0 \leq i \leq N-1$. A 2^b permutation of N elements sends the element from position i to position $i + 2^b \bmod N$. Assuming a SIMD hypercube which restricts all communications to a single dimension at a time, we have shown that $n - b$ routing steps are needed to perform a 2^b permutation, even if we allow any arbitrary one-to-one mapping of elements to processors [3].

We consider a class of parallel computations called $\pm 2^b$-ascend. A computation in this class may be expressed as follows:

```
1. For $b := 0$ to $n - 1$ do
2.     $A[i] \leftarrow f_{b,i}(A[i], A[i+2^b \bmod N], A[i-2^b \bmod N])$, $(0 \leq i \leq N-1)$
```

(Statement 2 is performed for all i in parallel.) In the light of the above lower-bound result, a straightforward implementation of this class on a SIMD hypercube would require $O(n^2)$ routing steps. Here, we outline an $O(n^2/\log n)$ algorithm, assuming $O(\log n)$ space per PE.

(In [2], we obtained an $O(n)$ algorithm for a similar class called $\pm 2^b$-descend. A computation in this class has the same structure as the above except the iterations run in the decreasing order of b. Batcher's odd-even merge falls in this class. The *descend* and *ascend* classes of computations defined previously by Preparata-Vuillemin [5] are subclasses of our $\pm 2^b$-descend and $\pm 2^b$-ascend, respectively.)

We first show that, given a sequence $a[0 : N-1]$, if each PE[i] contains two elements $a[i]$ and $a[i + 2^{b+1} \bmod N]$, then every PE[i] can acquire $a[i + 2^b \bmod N]$ using a single routing step across dimension b. (We refer to this as "midpoint acquisition".)

The basic idea of our ascend algorithm is to bring enough data into each PE to locally perform *several iterations.* Let $A[i]$ be a register in PE$[i]$. Let $A[i] = a^b[i]$ at the beginning of iteration b. The set of elements that effect the value of $A[i]$ during iterations $b = s, s+1, \ldots, s+j-1$ are the following $2^{j+1}-1$ elements (which are centered around i and are 2^s-spaced):

$$L_i^{s,s+j-1} = (a^s[i-2^s(2^j-1)], \ldots, a^s[i-2^s], a^s[i], a^s[i+2^s], \ldots, a^s[i+2^s(2^j-1)]).$$

(All arithmetic on PE indices are modulo N.) Our algorithm performs a group of j iterations (starting with iteration $b = s$) as follows:

1. Peform 2^{s+j} and -2^{s+j} permutations to bring $a^s[i-2^{s+j}]$ and $a^s[i+2^{s+j}]$ into PE$[i]$. This is done optimally in $2(n-s-j)$ routing steps.
2. By repeated applications of midpoint-acquisition, each PE$[i]$ obtains the sequence $L_i^{s,s+j-1}$. This is done with $2(2^j-1)$ routing steps.
3. Now each PE contains the data needed to perform j iterations locally. This takes $2(2^j-1)-j$ local-computational steps.

Thus, a group of $j = \log n$ iterations are run in $O(n)$ routing steps and $O(n)$ computational steps. The entire algorithm is run using $\lceil \frac{n}{\log n} \rceil$ such groups, with $O(\frac{n}{\log n} \times n)$ total complexity. See [3] for details.

Note

More recently, we have developed two new algorithms for the ascend class on a SIMD hypercube [4]. The first algorithm runs in $O(n^{1.5})$ time and requires $O(1)$ space per PE. The second algorithm runs in $O(n\sqrt{n/\log n})$ time and requires $O(\log n)$ space per PE. However, for practical values of n, the algorithm presented here is more efficient.

References

[1] S.L. Johnsson, "Communication Efficient Basic Linear Algebra Computations on Hypercube Architectures," *Journal of Parallel and Distributed Computing*, vol.4, pp. 133-172, 1987.

[2] D. Nassimi and Y.D. Tsai, "Efficient Implementations of a Class of $\pm 2^b$ Parallel Computations on a SIMD Hypercube," in *Proceedings of The 5th International Parallel Processing Symposium*, pp. 2-9, April 1991.

[3] D. Nassimi, "A Parallel Algorithm for the Class of $\pm 2^b$ ASCEND Computations on a SIMD Hypercube," New Jersey Institute of Technology, CIS Department, Technical Report CIS-91-13, June 1991.

[4] D. Nassimi, "Nearly Logarithmic-Time Parallel Algorithms for the Class of $\pm 2^b$ ASCEND Computations on a SIMD Hypercube," New Jersey Institute of Technology, CIS Department, Technical Report CIS-91-19, July 1991.

[5] F. Preparata and J. Vuillemin, "The Cube-connected Cycles: A Versatile Network for Parallel Computation," *Comm. of the ACM*, vol. 24, pp. 300-309, 1981.

Some Doubly Logarithmic Time Optimal Parallel Algorithms

Zhaofang Wen*
Department of Computer Science
University of Minnesota-Duluth
320 Heller Hall
Duluth, Minnesota 55812

Abstract

A problem is *highly parallelizable* if it can be solved by an optimal parallel algorithm of $O(\log\log n)$ time on a PRAM. We show that several problems are highly parallelizable. To solve these problems, we discover some interesting relations between data structures.

1 Introduction

A problem is *highly parallelizable* if it can be solved by an optimal parallel algorithm of doubly logarithmic time ($O(\log\log n)$) on a PRAM. Compared with the number of problems that can be solved by optimal parallel algorithms in $O(\log n)$ time, there is an extremely small number of problems known to be highly parallelizable [3]. The class of doubly logarithmic time optimal parallel algorithms and the challenge of designing such algorithms is discussed in [3]. In this paper, we further investigate this class. In particular, we solve the following problems in $O(\log\log n)$ time using $O(\frac{n}{\log\log n})$processors on the Common CRCW PRAM (which is the weakest among the CRCW PRAM models). (Refer to [10] for more details about the PRAM computation models).

- *The All Closest Enclosing Parentheses Problem*: Given a well-formed sequence of left and right parentheses in which every parentheses knows its mate. For each matched pair of parentheses, find the closest (inner most) pair of parentheses that encloses it.
- *The Tree Reconstruction Problem*: Reconstruct an ordered tree from its preorder and postorder traversals. (This is different from reconstructing a binary tree from its traversals.)
- *The Lowest Common Ancestor Problem*: An ordered tree T (along with its preorder and postorder traversals) is given for preprocessing. The preprocessing should enable one processor to find in constant time the Lowest Common Ancestor (LCA) of any pair of vertices in the tree.

To solve the problems above, we discover some interesting relations between data structures (which we beleive have much more applications beyond those discussed here). They include:

---

*Part of the research was performed when the author was with the Department of Computer Science, Old Dominion University, Norfolk, VA

- A data structure called the *R-forest* is defined as follows: The *R-forest* of an array $A = (a_1, a_2, ..., a_n)$ (denoted *R-forest*(A) or *R-forest*($a_1, ..., a_n$)) of n distinct real numbers is a forest with vertices labeled by the numbers. For each a_i, $1 \leq i \leq n$, the parent of a_i is the nearest element to its right that is smaller than a_i. We show that the well-known data structure *Cartesian tree* [17] is actually the binary tree representation of the *R-forest*, according to the *left-son-right-sibling* method [12]. (This relationship explains an observation about the *Range Minima* problem by *Berkman, Schieber and Vishkin* [2].)

- Let F be a forest (of ordered trees) with vertices labeled by n distinct numbers (e.g. $1, 2, ..., n$); and $c_1, c_2, ..., c_n$ and $d_1, d_2, ..., d_n$ be the preorder and postorder traversals of F, respectively. If $c_1, c_2, ...c_n$ is sorted, then $F =$*R-forest*($d_1, d_2, ..., d_n$). As one of its applications, this relationship suggests an alternative approach (compared to the approach to be presented in Section 4) to reconstruct an ordered tree from its preorder and postorder traversals.

1.1 Literature

It is well-known that a binary tree is uniquely determined by its preorder and inorder traversals [12]. Previously, the problem of reconstructing a binary tree from its inorder and preorder traversal was solved in [6, 3, 1, 14]. In particular, the algorithms in [6, 1] take $O(n)$ sequential time; the optimal algorithm in [3] runs in $O(\log\log n)$ time on a CRCW PRAM; and the optimal algorithm in [14] takes $O(\log\log n)$ time using optimal number of processors on a CREW PRAM. It is also known that an ordered tree can be uniquely represented as a binary tree using the *left-son-right-sibling* method [12]. The ordered tree and the corresponding binary tree have the same preorder traversal; and the postorder traversal of the ordered tree is the same as the inorder traversal of the binary tree. Therefore, an ordered tree is uniquely determined by its preorder and postorder traversals. However, the problem of reconstructing an ordered tree considered in this paper is not the same problem as reconstructing a binary tree [3, 14]. Because for an ordered tree, we cannot simply reconstruct its binary tree representation, we also need to determine the parent (in the ordered tree) of every vertex. Therefore, the algorithms in [3, 14] do not apply here.

The LCA preprocessing problem was considered in [9, 15, 5]. Specifically, the algorithm in [9] runs in O(n) sequential time. The algorithm in [15] runs in $O(\log n)$ time using $O(\frac{n}{\log n})$ processors on an EREW PRAM. The algorithm in [5] takes $O(\alpha(n))$ (here, $\alpha(n)$ stands for the inverse *Ackerman* function) time using an optimal number of processors on a CRCW PRAM under the following assumptions: the *Euler tour* (a notation due to [16]) of the tree is given in an already ranked form; the level of each vertex in the tree is also given. Without this assumptions, the algorithm in [5] runs in $O(\log n)$ time using an optimal number of processors.

An Observation of Berkman, Schieber, and Vishkin([2]): Let $A = (a_1, a_2, ...a_n)$ be an array of n distinct real numbers. Define R-ANSV(A) to be an array of indices as: For $1 \leq i \leq n$, R-ANSV(i)=j if the a_j is the nearest element to the right of a_i that is smaller than it. Define similarly RANGE-MINIMA(A) to be a matrix of size $n * n$ of indices. That is, in location (i, j) of the matrix we will have the index of the minimal element in the subarray $[a_i, a_{i+1}, ..., a_j]$. Then, for any arrays $A[1..n]$, and $B[1..n]$ of real numbers, R-ANSV(A) = R-ANSV(B) *iff* RANGE-MINIMA(A) = RANGE-MINIMA(B).

We notice that the relationship (more appropriately, the bridge) between the *R-forest* and the *Cartesian tree* explains quite well the observation above: R-ANSV(A) is uniquely determined the by the structure of the *R-forest*(A) (i.e. the structure of *Cartesian-tree*(A)). According to [7], the mimima in the subarray $[a_i, a_{i+1}, ..., a_j]$ is the LCA of a_i and a_j in *Cartesian-tree*(A), so the structure of the Cartesian tree also uniquely determines RANGE-MINIMA(A).

2 Basics

In this section, we list some basic tools which will be useful for our presentation in the following sections.

Definition 2.1 *(Vuillemin [17]) the* Cartesian tree *of an array* $A[1..n] = (a_1, a_2, ..., a_n)$ *of* n *distinct real numbers is a binary tree with vertices labeled by the numbers. The root has label* a_m, *where* $a_m = min\{a_1, a_2, ..., a_n\}$. *Its left subtree is a* Cartesian tree *for* $A[1..m-1] = (a_1, a_2, ..., a_{m-1})$ *and its right subtree is a* Cartesian tree *for* $A[m+1..n] = (a_{m+1}, a_2, ..., a_n)$. *(The tree for an empty sub-array is null.)*

We use *Cartesian-tree*(A), or *Cartesian-tree*($a_1, a_2, ..., a_n$) to denote the *Cartesian tree* of array $A = (a_1, a_2, ..., a_n)$.

The *All Nearest Smaller Values* (ANSV) problem ([3, 11]): Given an array $A = (a_1, a_2, ..., a_n)$ of n elements from a totally ordered domain. For each a_i, $1 \leq i \leq n$, find the nearest element to its left and the nearest element to its right, that are less than a_i, if such elements exist. That is, for each $1 \leq i \leq n$, find the maximal $1 \leq j < i$, and the minimal $i < k \leq n$ such that $a_j < a_i$ and $a_k < a_i$. We say that a_j is the *left match* and a_k is the *right match* of a_i.

The *Range Minima Search* problem: Given an array of n real numbers $A[1..n] = (a_1, a_2, ..., a_n)$, preprocess A. After the preprocessing, one processor can find in constant time the minimum element in a sub-array $A[i, j]$, for any $1 \leq i \leq j \leq n$.

Both the ANSV problem and the Range Minima Search problem can be solved optimally in $O(\log\log n)$ time using $O(\frac{n}{\log\log n})$ processors on a CRCW PRAM [4, 3], or in $O(\log n)$ time using $O(\frac{n}{\log n})$ processors on an EREW PRAM [11].

A binary tree representation of the trees, called the *left-son-right-silbing* method [12], is as follows: Let $F = (T_1, T_2, ..., T_n)$ be a forest of trees (ordered). The binary tree $B(F)$ corresponding to F can be defined as: (1) If $n = 0$, $B(F)$ is empty. (2) If $n > 0$, the root of $B(F)$ is *root*(T_1); the left subtree of $B(F)$ is $B(T_{11}, T_{12}, ..., T_{1m})$, where $T_{11}, T_{12}, ..., T_{1m}$ are the subtree of *root*(T_1) ; and the right subtree of $B(F)$ is $B(T_2, ..., T_n)$. It is shown in [12] that F and $B(F)$ have the same preorder traversal, while the inorder traversal of $B(F)$ is the same as the postorder traversal of F [12].

When solving problems given as input the preorder traversal $c_1, c_2, ...c_n$ and postorder (or sometimes, inorder) traversal $d_1, d_2, ..., d_n$ of a tree, we will always assume a special case in which $c_1, c_2, ...c_n$ is $1, 2, ..., n$. The general case where the preorder traversal is a permutation of $1, 2, ..., n$ can be reduced to the special case by changing the labeling of the tree. In particular, for a node with label j, we can change its label to $LF(j)$ defined as follows: $LF : \{1, 2, ..., n\} \rightarrow \{1, 2, ..., n\}$; $LF(j) = i$ if $c_i = j$. With the new labeling, the preorder traversal of the tree will be $1, 2, ..., n$, and its postorder traversal will be $LF(d_1), LF(d_2), ..., LF(d_n)$. It is important to notice that the cost to make such a reduction does not affect the overall complexity of our solutions, because changing (back and forth) the labels of a tree in this way can be done in O(1) time using n processors on a EREW PRAM (since we can implement LF using an array $LF[1..n]$ in an obvious way).

3 The All Closest Enclosing Parentheses Problem

The algorithm presented here will be needed in the next section to determine the parent of each node in a tree when we study the problem of reconstructing an ordered tree from its preorder and postorder traversals.

The *All Closest Enclosing Parentheses* problem:

Input: $A[1..n]$, an array containing a well-formed sequence of left and right parentheses in which every parenthesis knows the position of its mate.

Output: For each matched pair of parentheses in A, find the closest (inner most) pair of parentheses that encloses it. (We say a pair of parentheses $(A[i], A[l])$ encloses the other pair $(A[j], A[k])$ if $1 \leq i < j < k < l \leq n$.)

The algorithm consists of two step:

Step1: Build an array $B[1..n]$: For each $1 \leq i \leq n$, $B[i] = j$ such that $A[j]$ is the mate of $A[i]$ (i.e. $B[i]$ remembers the position of the mate of $A[i]$);

Step2: For each $1 \leq i \leq n$, if $A[i]$ is a left parenthesis, then find the maximal $1 \leq j < i$ such that $B[j] > B[i]$.

Theorem 3.1 *The* All Closest Enclosing Parentheses *problem can be solved optimally in $O(\log\log n)$ time using $O(\frac{n}{\log\log n})$ processors on a CRCW PRAM, or in $O(\log n)$ time using $O(\frac{n}{\log n})$ processors on an EREW PRAM.*

Proof. (sketch) To see the correctness, we notice that after Step 2, each left parenthesis (and therefore its mate) has found the closest pair of parentheses that encloses it. Using the parallel solutions for the ANSV problem [3, 11], Step 2 can be done in $O(\log\log n)$ time using $O(\frac{n}{\log\log n})$ processors on on a CRCW PRAM, or in $O(\log n)$ time using $O(\frac{n}{\log n})$ processors on an EREW PRAM. Step 1 can be done in $O(1)$ time using n processors. □

4 The Tree Reconstruction Problem

In this section, we show how to reconstruct an ordered tree from its preorder and postorder traversals.

It is well-known that an ordered tree can be encoded by a well-formed parentheses string. Let $\tilde{T}$ be the string representing an ordered tree T. The encoding is defined as follows: $\tilde{T} = \phi$ if T is an empty tree; otherwise, $\tilde{T} = (\tilde{T}_1\tilde{T}_2...\tilde{T}_k)$, where $T_1, T_2, ..., T_k$ are the subtrees of T.

By replacing each pair of matched left and right parentheses in the encoding above with two copies of a node label, we have a scheme as follows: $\tilde{T} = \phi$ if T is an empty tree; otherwise, $\tilde{T} = r\tilde{T}_1\tilde{T}_2...\tilde{T}_k r$, where r is the root of T. This encoding gives a sequence of labels, in which there are exactly two copies of each distinct label. We call this sequence the *p-p path* of T, since it has a nice property: keeping only the first copy of each label results in the preorder traversal of T, while keeping the second copy of each label yields a postorder traversal of T. Moreover, we have the following observation on the *p-p path*:

Observation 4.1 *Let u and v be two nodes in the tree. u is an ancestor of v in T if and only if the pair of u encloses the pair of v in the* p-p path. *u is the right sibling of v in the ordered tree if and only if the first copy of u appears next to the right of the second copy of v in the* p-p path.

Lemma 4.1 *A sequence of labels $b_1, b_2, ..., b_{2n}$ represents a* p-p path *of an n-node ordered tree* T if and only if *the following conditions hold: (1)$b_1 = b_{2n}$; (2) there are exactly two copies of each label in the sequence; and (3) there exist no indices $1 \leq i < j < k < m \leq 2n$ such that $(b_i = b_k)$ and $(b_j = b_l)$.*

Proof (sketch). By the definition of the *p-p path*, the "only if" part of the lemma is obvious. The "if" part can be shown by induction on n. □

Our algorithm to reconstruct an ordered tree from its preorder and postorder traversals consists of two steps:

Algorithm Traversals-to-ordered-tree;

Step 1: Construct the *p-p path* in which each label knows the position of its duplicate.

Step 2: Reconstruct the ordered tree from the *p-p path.*

Step 2 is to reconstruct the ordered tree from the *p-p path.* In other words, Step 2 is to determine, for each node, its left and right siblings and its parent in the ordered tree. By Observation 4.1 and Theorem 3.1, Step 2 can be implemented in $O(\log\log n)$ time using $O(\frac{n}{\log\log n})$ processors on a CRCW PRAM, or in $O(\log n)$ time using $O(\frac{n}{\log n})$ processors on an EREW PRAM.

We now focus on Step 1, i.e. constructing the *p-p path* from the traversals. We implement Step 1 using a similar technique in [14], where the technique is used to construct a so-called *preorder-inorder path* from the preorder and the inorder traversals of a binary tree. The main idea here is to reduce the problem of computing the *p-p path* of the tree from the traversal to the problem of merging two sorted sequences. It is described as follows:

Let $c_1, c_2, ...c_n$ and $d_1, d_2, ..., d_n$ be the preorder and postorder traversals of an ordered tree T. For simplicity, we assume that $c_1, c_2, ...c_n$ is $1, 2, ..., n$. Construct two sequences of triples: a sequence $(1, j_1, c_1), (1, j_2, c_2), ..., (1, j_n, c_n)$ such that $d_{j_i} = c_i, (i = 1, 2, ..., n)$ (i.e. j_i is the position of c_i in sequence $d_1, d_2, ..., d_n$); and a sequence $(2, 1, d_1), (2, 2, d_2), ..., (2, n, d_n)$. Denote $\prod = \{(1, j_1, c_1), (1, j_2, c_2), ..., (1, j_n, c_n), (2, 1, d_1), (2, 2, d_2), ..., (2, n, c_n)\}$ Defined a binary relation $\ll$ on $\prod$ as follows: for arbitrary triples (α, β, γ) and $(\alpha', \beta', \gamma')$ in $\prod$ we have:

Rule 1. $((\alpha = 1) \wedge (\alpha' = 1)) \rightarrow (((\alpha, \beta, \gamma) \ll (\alpha', \beta', \gamma')) \leftrightarrow (\gamma < \gamma'))$;

Rule 2. $((\alpha = 2) \wedge (\alpha' = 2)) \rightarrow (((\alpha, \beta, \gamma) \ll (\alpha', \beta', \gamma')) \leftrightarrow (\beta < \beta'))$;

Rule 3. $((\alpha = 1) \wedge (\alpha' = 2)) \rightarrow (((\alpha, \beta, \gamma) \ll (\alpha', \beta', \gamma')) \leftrightarrow ((\beta < \beta') \vee (\gamma \leq \gamma')))$

Rule 4. $((\alpha = 2) \wedge (\alpha' = 1)) \rightarrow (((\alpha, \beta, \gamma) \ll (\alpha', \beta', \gamma')) \leftrightarrow ((\beta < \beta') \wedge (\gamma < \gamma')))$

By Rules 1-2, both sequences $(1, j_1, c_1)$, $(1, j_2, c_2)$, ..., $(1, j_n, c_n)$ and $(2, 1, d_1)$, $(2, 2, d_2)$, ..., $(2, n, d_n)$ are already sorted according to $\ll$. Merging these two sequences according to $\ll$ we obtain a sequence of triples: $(\alpha_1, \beta_1, \gamma_1)$, $(\alpha_2, \beta_2, \gamma_2)$, ..., $(\alpha_n, \beta_n, \gamma_n)$.

Claim 4.1 *γ_1 , γ_2, ..., γ_n is the* p-p path *of the tree determined by the traversals.*

Proof. (sketch) The correctness of the claim relies on the following facts: (a)There are exactly two copies of each label in $\gamma_1, \gamma_2, ..., \gamma_n$, and $\gamma_1 = \gamma_{2n}$ (satisfying the conditions (1)-(2) of Lemma 4.1); (b) There do not exist integers $1 \leq i < j < k < l \leq 2n$ such that $\gamma_i = \gamma_k$ and $\gamma_j = \gamma_l$ (satisfying the condition (3) of Lemma 4.1); (c) Deleting the second copies of the duplicate labels in $\gamma_1, \gamma_2, ..., \gamma_n$ results in $c_1, c_2, ..., c_n$, and deleting the first copies of the duplicate labels in in $\gamma_1, \gamma_2, ..., \gamma_n$ gives $d_1, d_2, ..., d_n$. □

Up to now, we have reduced computing the *p-p path* to merging two sorted sequences. Using the $O(\log\log n)$ time optimal merging algorithm in [13], or the $O(\log n)$ time optimal merging algorithm in [8], we have,

Theorem 4.1 *An ordered tree can be reconstructed from its preorder and postorder traversals in $O(\log\log n)$ time using $O(\frac{n}{\log\log n})$ processors on a CRCW PRAM, or in $O(\log n)$ time using $O(\frac{n}{\log n})$ processors on an EREW PRAM.* □

5 Some Relations Between Data Structures

Before we proceed to solve the LCA problem in the next section, we discuss the relations between several data structures. The following data structure is motivated by the ANSV problem.

Definition 5.1 *The* R-forest *of an array $A = (a_1, a_2, ..., a_n)$ of n distinct real numbers (denoted* R-forest(A)*, or* R-forest($a_1, a_2, ..., a_n$)*) is a forest with vertices labeled by the numbers. For each a_i, $1 \leq i \leq n$, the parent of a_i (if exists) is the element a_k with the minimal index $i < k \leq n$ such that $a_k < a_i$. In other words, the parent of a_i is the nearest element to its right that is smaller than a_i.*

The *R-forest* defines (in an obvious way) an ordered forest (i.e. a forest of ordered trees). It is easy to see that *R-forest*$(a_1, a_2, ..., a_n)$ can be constructed by solving the ANSV problem on $(a_1, a_2, ..., a_n)$.

Theorem 5.1 *For any array* $A = (a_1, a_2, ..., a_n)$ *of* n *distinct real numbers,* Cartesian-tree(A) *is the binary tree representation of* R-forest(A) *according to the* left-son-right-sibling *method.*

Proof (By induction on n) (In the proof, $B(F)$ denote the binary tree representation of a forest F). When $n = 1$ the argument is true. Assume the argument is true for $n < l$. When $n = l$, let $a_m = min\{a_1, a_2, ..., a_n\}$ and $(T_1, T_2, ..., T_j)$ =*R-forest*(A). By definition, a_m must be the root of T_1; the subtrees of a_m is *R-forest*$(a_1, a_2, ..., a_{m-1})$; and $(T_2, ..., T_j)$ =*R-forest*$(a_{m+1}, ..., a_n)$. By the induction hypothesis, we have $B($*R-forest*$(a_1, ..., a_{m-1}))$ =*Cartesian-tree*$(a_1, ..., a_{m-1})$, and $B($*R-forest*$(a_{m+1}, ..., a_n))$ =*Cartesian-tree*$(a_{m+1}, ..., a_n)$. Therefore, $B($*R-forest*$(A))$=*Cartesian-tree*(A) according to the *left-son-right-sibling* method. □

Theorem 5.2 *Let* F *be a forest (of ordered trees) with vertices labeled by* n *distinct numbers (e.g.* $1, 2, ..., n$*); and* $c_1, c_2, ..., c_n$ *and* $d_1, d_2, ..., d_n$ *be the preorder and postorder traversals of* F*, respectively. If* $c_1, c_2, ...c_n$ *is sorted (in increasing order), then* F =R-forest$(d_1, d_2, ..., d_n)$.

Proof. Again, we prove the theorem by induction on n. When $n = 1$ the argument is true. Assume that the argument is true for $n < l$. When $n = l$, let $F = (T_1, T_2, ..., T_j)$. Assume $d_i = c_1$. The preorder and postorder traversals of T_1 must be $c_1, ...c_i$ and $d_1, ..., d_i$, respectively. The preorder and postorder traversals of forest $(T_2, ..., T_j)$ are $c_{i+1}, ...c_n$ and $d_{i+1}, ..., d_n$, respectively. By Definition 5.1, *R-forest*$(d_1, ..., d_i)$ is a tree since $d_i = c_1 = min\{d_1, d_2, ..., d_n\}$. By the induction hypothesis, *R-forest*$(d_1, ..., d_i) = T_1$ and *R-forest*$(d_{i+1}, ..., d_n) = (T_2, ..., T_j)$. Therefore, F =*R-forest*$(d_1, d_2, ..., d_n)$. □

From Theorems 5.1 and 5.2, we have,

Corollary 5.1 *If* T *is a binary tree with preorder traversal* $1, 2, ..., n$ *and inorder traversal* d_1*,* d_2*, ...,* d_n*, then* T =Cartesian-tree$(d_1, d_2, ..., d_n)$. □

To see the applications of these relations between data structures, we notice that Theorem 5.2 and Corollary 5.1 suggest alternative approaches (different from the ones in [3, 14] and the one in Section 4) to reconstruct binary trees and ordered trees from their traversals.

6 The Lowest Common Ancestor (LCA) Problem

In this section, we solve the LCA problem: An ordered tree T along with its preorder and postorder traversals is given for preprocessing. After the preprocessing, one processor should be able to find in $O(1)$ time the lowest common ancestor of any pair of vertices in T.

Observation 6.1 *Let* u *and* v *be two vertices of* T*.* u *is an ancestor of* v *if and only if* u *appears before* v *in the preorder traversal and after* v *in the postorder traversal.* □

Observation 6.2 *Let T be an ordered tree and $B(T)$ be its binary tree representation; u and v be two vertices of T; w be the LCA of u and v in $B(T)$; $P(w)$ be the parent of w in T. Then in T the LCA of u and v is either w or $P(w)$.* □

With these observations, we are ready to show how to solve the LCA preprocessing problem. Let $c_1, c_2, ..., c_n$ and $d_1, d_2, ..., d_n$ be the preorder and postorder traversal of T, respectively. We assume that $c_1, c_2, ..., c_n$ is $1, 2, ..., n$ (if not, we can change the labeling of T using the method presented in Section 2 so that under the new labeling the preorder traversal of T is $1, 2, ..., n$).

Preprocessing: Solve the Range Minima Search problem on $(d_1, d_2, ..., d_n)$.

The preprocessing step can be done in $O(\log\log n)$ time using $O(\frac{n}{\log\log n})$ processors on a CRCW PRAM with a result in [3].

Let $B(T)$ be the binary tree representation of T according the *left-son-right-sibling* method. By Corollary 5.1, $B(T)$ =*Cartesian-tree*$(d_1, d_2, ..., d_n)$, since $d_1, d_2, ..., d_n$ (the postorder traversal of T) is also the inorder traversal of $B(T)$.

Query processing: Let u and v be two vertices of T. Compute w, the range minima between u and v in $(d_1, d_2, ..., d_n)$. By a property of the Cartesian tree [7], w is the LCA of u and v in $B(T)$. Let $P(w)$ be the parent of w in T. Check (according to the condition in Observation 6.1) whether w is an ancestor of both u and v in T; if not then $P(w)$ must be the LCA of u and v in T. (The query processing step can be done in constant time by one processor.)

Theorem 6.1 *Given an ordered tree along with its preorder and postorder traversals, a preprocessing can be done in $O(\log\log n)$ time using $O(\frac{n}{\log\log n})$ processors on a CRCW PRAM, so that after the preprocessing one processor can find in $O(1)$ time the lowest common ancestor of any pair of vertices in the tree.* □

Acknowledgement: Thanks to *Omer Berkman* for commenting on an early draft of this paper which improved the presentation, and telling me the observation about the range minima.

References

[1] A. Anderson and S. Carlsson. Construction of a tree from its traversals in optimal time and space. *Information Processing Letters*, 34:21–25, 1990.

[2] O. Berkman. Private communication.

[3] O. Berkman, D. Breaslauer, Z. Galil, B. Schieber, and U. Vishkin. Highly parallelizable problems. In *Proceedings of the Annual IEEE Symposium on Foundation of Computer Science*, pages 309–319, 1989.

[4] O. Berkman, B. Schieber, and U. Vishkin. Some doubly logarithmic parallel algorithms based on finding all nearest smaller values. *UMIACS-TR-88-79*, University of Maryland institute for advanced computer studies, 1988.

[5] O. Berkman and U. Vishkin. Recursive star-tree parallel data-structure. In *Proceedings of the Annual ACM Symposium on Theory of Computing*, 1990.

[6] G. H. Chen, M. S. Yu, and L. T. Liu. Two algorithms for constructing a binary tree from its traversals. *Information Processing Letters*, 28:297–299, 1988.

[7] H. N. Gabow, J. L. Bentley, and R. E. Tarjan. Scaling and related techniques for geometry problems. In *Proceedings of the Annual ACM Symposium on Theory of Computing*, pages 135–143, 1984.

[8] T. Hagerup and C. Rub. Optimal merging and sorting on the erew pram. *Information Processing Letters*, 33:181–185, 1989.

[9] D. Harel and R. E. Tarjan. Fast algorithms for finding nearest common ancestor. *SIAM J. Comput.*, pages 338–355, 1984.

[10] R. M. Karp and V. Ramachandran. A survey of parallel algorithm for shared memory machines. *Rep. No. UCB/CSD 88/408*, Computer Science Division, University of California, Berkeley, CA 94720, 1988.

[11] S. K. Kim. Optimal parallel algorithm on sorted intervals. In *Proceedings of the 27th Annual Allerton Conference on Control, Communication, and Computing*, 1989.

[12] D. E. Knuth. *The Art of Computer Programming, Vol. 1, Fundamental Algorithms.* Second Edition, Addison-Wesley, Reading, Mass., 1973.

[13] C. P. Kruskal. Searching, merging, and sorting in parallel computation. *IEEE Transactions on Computers*, 32:924–946, 1983.

[14] S. Olariu, C. M. Overstreet, and Z. Wen. An optimal parallel algorithm to reconstruct a binary tree from its traversals. In *Proceedings of the International Conference on Computing and Information, Springer-Verlag*, 1991.

[15] B. Schieber and U. Vishkin. On finding lowest common ancestor: simplification and parallelization. *SIAM J. Comput.*, pages 1253–1262, 1988.

[16] R. E. Tarjan and U. Vishkin. An efficient parallel biconnectivity algorithm. *SIAM J. Comput.*, pages 862–874, 1985.

[17] J. Vuillemin. A unified look at data structures. *Comm. of ACM*, 1980.

Parallel Algorithms for Arithmetic Expression Parsing *

Weian Deng & S. Sitharama Iyengar
Department of Computer Science
Louisiana State University
Baton Rouge, LA 70803

Abstract

The advent of commercial multiprocessor has sparked considerately interest in the study and development of parallel algorithms. A major issue in the design of fast parallel algorithms is to identify techniques that yield algorithms that are both fast and make an efficient use of processors. Towards this objective, this paper presents an efficient expression parsing parallel algorithm using SIMD-SM-R model which has a time complexity of O(log n), where n is the length of the input string.

Also, we present two optimal expression parsing algorithm using SIMD-SM model of computation. One of them has a time complexity of O($\log^2 n$) (Comparable to Dekel and Sahni's) with an optimal bound on the EPU cost. A comparison of our algorithm with the other existing algorithms are presented.

1.0 INTRODUCTION

As parallel computing machines become more and more widely used, the need for efficient parallel parsing technique has drawn more and more attention. In this paper, we focus on how to parse arithmetic expressions efficiently. By parsing, we mean to get a tree form representation from an infix expression. Many papers have been written on this topic.[DeSa 83], [BaVi 85] and [Srik 90] are some of the important results.

Dekel and Sahni [DeSa 83] considered the generation of postfix form from infix form for an given arithmetic expression of length n, and the translation of postfix form into tree form.Originating from the standard priority-based sequential infix to postfix algorithm, they used stable sort technique to pair the matching parentheses together, so that information can be exchanged among the tokens in a expression in parallel. They came up with two parallel algorithms on an SIMD-SM model. One used n processors and O($\log^2 n$) time, and the other used $n^2/\log n$ processors and O(log n) time. Neither is optimal by means of Effectiveness of Processor Utilization(EPU in short).

Bar-On and Vishkin[BaVi 85] presented the best parallel algorithm so far for obtaining the tree form of an arithmetic expression. Recognizing that postfix form is not so useful in parallel evaluation of arithmetic expressions, they transformed infix form to tree form directly, and came up with an optimal parallel algorithm with O(log n) time complexity using $n/\log n$ processors. The heart of their algorithm consists of the development of an efficient parenthesis pairing algorithm in O(log n) time with $n/\log n$ processors, and the introduction of the concept of simple expressions. However, their algorithm is based on the SIMD-SM-R model which is more powerful than that used by Dekel and Sahni. [DeSa 83]

Srikant [Srik 90] introduced another parallel algorithm for expression parsing. But his focus was on developing algorithms running on the other kinds of computational models, like mesh-connected machine, shuffle-exchange machine, and cube-connected machine, etc.

---

* This research is supported partly by LEQSF-RD-A04 grant from LSU Board of Regents.

In this paper, beginning with Bar-On and Vishkin's ideas, we discuss three parallel algorithms for tree form generation of arithmetic expressions, one on SIMD-SM-R model, and the other two on SIMD-SM model. On the SIMD-SM model, we optimize the performance in terms of EPU. The main idea here is how to avoid the read conflict posted by Bar-On and Vishkin's algorithm by modifying their parenthesis pairing algorithm. In next section, we will introduce some necessary concepts and notations, and briefly describe Bar-On and Vishkin's algorithm. In section 3, a parallel algorithm for parenthesis pairing on SIMD-SM-R model is described which is simple and efficient. Afterwards, two optimal parenthesis pairing algorithms on SIMD-SM model are presented. Finally, in our conclusion, a comparison of the main results in expression tree form generation is given.

2.0 ARITHMETIC EXPRESSION PARSING on PARALLEL MODEL

The problem we concern here is how to construct a binary evaluation tree for a proper arithmetic expression of length *n*. The expression is constructed from constants, variables, parentheses and operators (+, -, *, / and ^). The operators follow the usual priority rule and associativities. All operators except ^ are left associative. Each proper arithmetic expression corresponds to exactly one binary tree representation, in which all the leaves correspond to the operands and all the internal nodes correspond to the operators of the expression. The connection between these nodes are determined by the priorities and the associativities.

2.1 Tree Form Generation Technique

One of the strategies useful to develop efficient parallel algorithms is to find some simpler sub-structures from which the more complicated structure we need can be formed. The tree form for an expression is a relatively complicated structure. Here, we are able to find simpler structures from which tree forms for arithmetic expressions can be constructed. They correspond to linear expressions and simple expressions.

Definition 1: A linear expression is an proper arithmetic expression which consists of constants, variables and a set of operators with the same priority.

For example, $a + b - 2 + c$ is a linear expression. But $a + 2 \times b$ and $a + (8 - c)$ are not. Linear expressions have the following important property.

Property 1: In the tree form of a linear expression, the nodes corresponding to the operators form a link in which the root node corresponds to the left most operator if the operators are right associated or to the right most operator if they are left associated.

An example of simple expression is shown in Figure 1. With linear expressions as infrastructures, any arithmetic expression can be constructed.

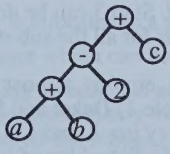


Figure 1. Tree Form of Linear Expression $a + b - 2 + c$

Definition 2: A sub-expression is a successive portion of a given arithmetic expression enclosed by a pair of matching parentheses.

Definition 3: A simple expression is an expression in which all operators within one sub-expression but not in any other sub-expression are of the same priority. [BaVi 85]

In other words, a simple expression is an arithmetic expression in which each sub-expression of it is a linear expression, provided that all the sub-expressions within the former mentioned sub-expression are considered as its operands. Therefore, any sub-expression of a simple expression can be seen as a linear expression as well as an operand for its upper level sub-expression. As an operand, each sub-expression can be represented by its root operator. In this way, the complicated tree form for an expression can be found by simply translating the expression to a simple expression, then finding all the tree forms for all the sub-expressions in the simple expression. For the following simple expression:

$$(a/b)+(c\wedge((d/e)+(f\wedge g)))$$

After finding all the sub-expressions,

$$e_1=(a/b) \qquad e_4=((d/e)+(f\wedge g))=e_2+e_3$$
$$e_2=(d/e) \qquad e_5=(c\wedge((d/e)+(f\wedge g)))=c\wedge e_4$$
$$e_3=(f\wedge g) \qquad e_6=\ (a/b)+(c\wedge((d/e)+(f\wedge g)))=e_1+e_5$$

we can construct the tree form for all the *e*'s

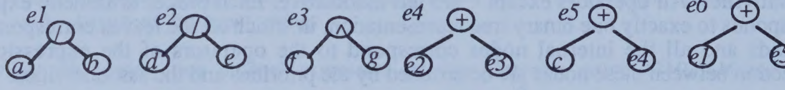


Figure 2. Tree Forms for All The Linear Sub-expressions

Since the root operators are considered as the labels of the sub-expressions, the tree form (See Figure 3) is formed automatically.

It is now straightforward to develope the parallel algorithm to generate tree form in this way:

1. Translate the given expression to a simple expression;
2. Allocate all the sub-expressions;
3. Construct the tree form for all the sub-expressions.

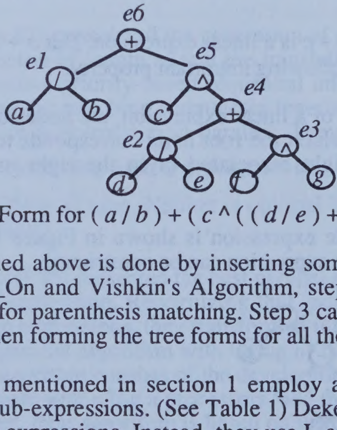


Figure 3. Tree Form for $(a/b)+(c\wedge((d/e)+(f\wedge g)))$

The first step mentioned above is done by inserting some more parentheses into the expression.[Knut 62] In Bar_On and Vishkin's Algorithm, step 2 is implemented by their parenthesis pairing algorithm for parenthesis matching. Step 3 can be done by finding the root for each sub-expression and then forming the tree forms for all the sub-expressions.

The three algorithms mentioned in section 1 employ almost the same idea: using parenthesis matching to find sub-expressions. (See Table 1) Dekel and Sahni [DeSa 83] didn't employ the concept of simple expressions. Instead, they use L and LU values. To find L and LU values, they find the sub-expressions in the form of a linked list of its operators.

Considering time complexity, all the steps except parenthesis pairing can be done in O(1) time in both Bar-On & Vishkin's algorithm and Srikant's algorithm, and O(log *n*) time in Dekel & Sahni's algorithm. We can see from Table 1 that the effectiveness of parenthesis pairing determines the effectiveness of the whole algorithm. So, in the rest of the paper, we discuss parenthesis pairing algorithms only.

Table 1. Summary of the Existing Algorithms

Investigations	Parenthesis Matching	Sub-Expression Finding	Time Complexity
Dekel & Sahni (1983)	Stable Sort	Using L & LU value	$O(\log^2 n)$
Bar-On & Vishkin (1985)	$O(\log n)$ Algorithm	Simple Expression	$O(\log n)$
Srikant (1990)	Stable Sort	Simple Expression	$O(\log^2 n)$

2.2 Bar-On and Vishkin's Parenthesis Pairing Algorithm

The problem of parenthesis pairing is that, given a proper parenthesis sequence of length $2n$, n is a positive integer, pair all the matching left parentheses with their right parentheses.

Suppose the input sequence is divided into m segments, where $n = mq$, for a certain positive integer q. Each segment has $2q$ parentheses. These segments are not necessary to be the proper ones. In each segment S_k, $1 \leq k \leq m$, there are two kinds of parentheses; one has all the parentheses which can be paired within the segment, and the other includes the parentheses in that segment unable to be paired. Removing all the paired parentheses from the segment, we can get a sequence of the form:

)) . . .)) ((. . ((

i successive right parentheses are followed by j successive left parentheses.

Definition 4: The sequence of parentheses formed by removing from a segment S_k of a proper parenthesis sequence all the paired parentheses within the segment. The remaining parenthese form an unpaired sequence in segment S_k which is denoted by U_k.

If matching parentheses have been found for all the left most left parenthesis (LMLP in short) and the right most right parentheses (RMRP in short) of all the unpaired sequences of a given proper sequence, and these paired parentheses are marked, there exists a very useful property. [BaVi 85]

Property 2: In each unpaired sequence U_k, $1 \leq k \leq m$ of a given proper parenthesis sequence, the sequence of left parentheses to the right of the LMLP and to the left of next marked left parenthesis can be paired one after an other by the sequence of right parentheses to the right of the corresponding right parenthesis of this LMLP.

[BaVi 85] has the proof for this property. In their algorithm, Bar-On and Vishkin divided the input parenthesis sequence of length n into $n/\log n$ segments S_k with length $\log n$ where each segment S_k is allocated to a processor P_k. The parallel algorithm has four steps:

1. The parentheses which can be paired within the segment S_k are found by sequential algorithm using a stack, and are removed by marking;
2. Construct a balanced binary tree with parenthesis nesting level information;
3. For the unpaired sequences U_k, $1 \leq k \leq n/\log n$, a variant of binary search on this binary tree is used to find the matching parentheses for the LMLPs and RMRPs of all the unpaired sequence U_k;
4. Property 2 is used to pair all the rest of the parentheses.

This parenthesis pairing algorithm has a time complexity of $O(\log n)$ and uses only $n/\log n$ processors. However, the computational model used is SIMD-SM-R, which allows the concurrent read operation. In the following section, based on the method of sequence segmentation, we will present our parallel algorithms for parenthesis pairing.

3.0 An NEW ALGORITHM on SIMD-SM-R

The idea for our first parenthesis pairing algorithm comes from our observation on the property of unpaired sequences. Given a parenthesis sequence of length *n*, we can first divide the sequence into *n* segment S_k, for $1 \leq k \leq n$, each of length 1 and unpaired. Then, concatenate all the pairs of segments S_k and S_{k+1}, for all odd numbers *k* satisfying that $1 \leq k \leq n$. The result is *n*/2 unpaired sequences of length at most 2 (some unpaired sequence may be empty). During this concatenation, the pairs of parentheses matched in each pair of segments are removed so that the unpaired sequences in every segment of length 2 are found. After this, using the same method, we concatenate the *n*/2 unpaired sequences to get *n*/4 unpaired sequences for segments of length 4. For example, Figure 4 is the parenthesis pairing process for a given parenthesis sequence. (*i*, *k*) in the figure labels the resulting unpaired sequence $U_{i,k}$ of the *k*th segment after *i*th concatenation. Each of the $n/2^i$ unpaired sequences in level *i* is at most of length 2^i.

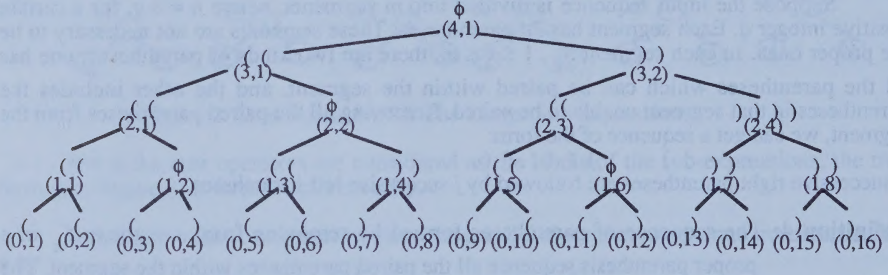


Figure 4. Pairing Process for a Given Parenthesis Sequence.

```
Algorithm 1 (SIMD-SM-R)
/* With $n/2$ processors, pair all the matching parentheses using divide and
   conquer technique */
1. For  $i$ from level 0 to level $\log n - 1$ do
2.     For all the sequences $U_{i,k}$ and $U_{i,k+1}$,  $1 \leq k \leq n/2^i$ ,
              $k$ is an odd number  do
3.         For all processors $j$ ($1 \leq j \leq 2^i$) assigned to sequences
                $U_{i,k}$ and $U_{i,k+1}$ do
4.             Reverse sequence $U_{i,k}$;
5.             If the $j$th parenthesis in the reverse of $U_{i,k}$ is '(' and
                     the $j$th parenthesis in$U_{i,k+1}$ is ')'
               Then pair them and remove them from $U_{i,k}$ and $U_{i,k+1}$;
6.             If the $j$-1st position contains a matching pair,
                     But in position $k$, no match is found
               Then recorder this position;
7.             Reverse $U_{i,k}$ back to its original location;
8.             Shift the remainder of $U_{i,k+1}$ to the right of the
                     remainder of $U_{i,k}$ in the position recorded in step 6;
9.         endfor
10     endfor
11. endfor
```

On SIMD-SM-R model, we use $n/2$ processors to realize the algorithm. On each level i, 2^i processors are used to concatenate $U_{i,k}$ and $U_{i,k+1}$, for each odd number k satisfying that $1 \le k \le n/\log n$, in order to find unpaired sequence $U_{i+1,\frac{k+1}{2}}$ on level $i + 1$. Because there are $n/2^i$ unpaired sequences on level i, totally, we need $(\frac{n}{2^i}/2) \times 2^i = n/2$ processors.

Because $U_{i,k}$ has the form `)) ...) (... ((' with r_k successive right parentheses followed by l_k successive left parentheses, $U_{i,k+1}$ has r_{k+1} successive right parentheses followed by l_{k+1} successive left parentheses. When concatenating them together, only the left parentheses in $U_{i,k}$ and the right parentheses in $U_{i,k+1}$ can possibly be paired. The number of matching pairs when $U_{i,k}$ and $U_{i,k+1}$ are concatenated is $\max(l_k, r_{k+1})$. Therefore, the strategy here is simply to assign a processor to each parenthesis in $U_{i,k}$, comparing the right most parenthesis in $U_{i,k}$ with the left most parenthesis in $U_{i,k+1}$, the second right most parenthesis in $U_{i,k}$ with the second left most parenthesis in $U_{i,k+1}$, . . ., etc. For each processor, if the parenthesis in $U_{i,k}$ is a left parenthesis and the parenthesis in $U_{i,k+1}$ is a right parenthesis, then pair these two parentheses, and remove them from the sequence. This can be done by the processor in O(1) steps. After that, in one step, these 2^i processors are used to concatenate the rest of the two sequences together to get another unpaired sequence in a higher level.

From the above discussion, a parallel algorithm can be described precisely as Algorithm1.

Lemma 1: In algorithm 1, after the ith iteration from step 2 to step 10, the $n/2^i$ sequences we get are exactly the unpaired sequences of the segments which divide the input proper sequence into $n/2^i$ successive sequences with the same length, and all the matching parentheses within each segment have been found.

Proof: We prove the lemma by induction on the number i of iterations performed.

$i = 0$ is the initial situation, in which we have $n/2^0 = n$ segments $S_{0,k}$, $1 \le k \le n$, Each segment $S_{0,k}$ has only one parenthesis. Because no matching parentheses exist in each segment , $U_{0,k} = S_{0,k}$.

Suppose for $i = m$, Lemma 1 is true. That is, for all $1 \le k \le n/2^m$, the unpaired sequence of segment $S_{m,k}$ is the same as $U_{m,k}$ found by the algorithm, and all the matching parentheses within each segment $S_{m,k}$ have been found.

When $i = m + 1$, for all $1 \le k \le n/2^{m+1}$, considering the sequence $U_{m+1,k}$ found by algorithm 1 after m+1st iteration from step 2 to step 10. It is constructed from $U_{m,2k-1}$ and $U_{m,2k}$. Also, we have

$$S_{m+1,k} = S_{m,2k-1} \circ S_{m,2k}$$

From the hypothesis, U\s\do6(m\,2k-1) and U\s\do6(m\,2k) are the unpaired sequences of segments $S_{m,2k-1}$ and $S_{m,2k}$ of the input. Moreover, every parentheses in $S_{m+1,k}$ which can be paired within $S_{m,2k-1}$ or $S_{m,2k}$ has been removed from $U_{m,2k-1}$ or $U_{m,2k}$ before the m+1st iteration. Since all the matching parentheses between $S_{m,2k-1}$ and $S_{m,2k}$ can be removed by executing step 2 to step 10 in the m+1st iteration, all the matching parentheses within $S_{m+1,k}$ are removed after the m+1st iteration. In addition, the sequence $U_{m,k+1}$ is formed from the reminding parenthesis sequence after removing all the matching parentheses in $S_{m+1,k}$. Therefore, $U_{m,k+1}$ is the unpaired sequence of $S_{m+1,k}$.

Theorem 1: Algorithm 1 can pair all the parentheses for a proper parentheses sequence of length n in O(log n) time using $n/2$ processors.

Proof: From Lemma 1, we have, after the ith iteration, the $n/2^i$ unpaired sequences of the $n/2^i$ segments of input, and all the matching parentheses within each segment of length 2^i have been paired. So, after log n iterations, we get $n/2^{\log n} = 1$ unpaired sequence of the only one segment $S_{\log n,1}$. Because the input is a proper sequence, every parentheses can be paired at this time and $U_{\log n,1}$ is empty.

For the time complexity, the algorithm executes step 2 to step 10 for exactly log n times. As we have discussed before, step 2 to step 10 can be done in constant time. So, the time complexity for algorithm 1 is O(log n) with $n/2$ processors.

Algorithm 1 is a good example of using the divide and conquer strategy to develop an efficient parallel algorithm. The logic of the algorithm is simple and easy to understand. Though it employs a large number of data moves, because the operations are highly parallel, the time complexity is low O(log n). Considering the space complexity, it needs one n element array to mark the paired parentheses, and another n element array to record the positions in step 6. Compared to Bar-On and Vishkin's algorithm, the time complexity are the same. But, our algorithm has a lower constant factor. This is because no binary tree is constructed in our algorithm. For the same reason, our algorithm has a lower space cost. However, it is not optimal.

4.0 OPTIMAL PARALLEL ALGORITHMS on SIMD-SM MODEL

Bar-On and Vishkin's algorithm is optimal on SIMD-SM-R model of computation. This model allows concurrent read operation to shared memory locations by multiple processors. To get an optimal algorithm on SIMD-SM model which allows exclusive read and exclusive write only, one straightforward way is to simulate the concurrent read operations on SIMD-SM by the technique described in[Akl 90]. In Bar-On and Vishkin's algorithm, read conflicts happen only when they use the binary tree to search for matching parentheses for LMLPs and RMRPs of all the unpaired sequences.

Algorithm 2 (SIMD-SM)

Suppose we use N processors to match a proper sequence of n parentheses using Bar-On and Vishkin's method. To simplify our discussion, without loss of generality, let $n = \log k$, for certain positive integer k .

Because the balanced binary tree has all the n parentheses as its leaves, the whole tree has $2n - 1$ nodes. All the data items in the nodes are possible to be read concurrently. So, for each such data item, we need to add $2N$-2 memory units to construct a binary tree with the data item as its root. This tree has N leaves corresponding to the N processors. By broadcasting the datum being read to all the leaves, read conflict is avoided. Totally, we need O(Nn) more space to solve the read conflict problem for each node containing a constant number of data items. In this way, every read operation on SIMD-SM-R is simulated on SIMD-SM in O(log N) times.

Theorem 2: Algorithm 2 has a time complexity t = O(n/N + (log N)(log n)) and a optimal cost c = O(n), for all $N \leq n/\log^2 n$.

Though this direct simulation has optimal performance and has an O($\log^2 n$) time complexity which is comparable to Dekel and Sahni's algorithm, it needs much more space to solve the read conflict problem. By observing the property of the unpaired sequence, we are able to get another optimal algorithm which uses much less space.

Let the proper parenthesis sequence input have length n. N processors are used. The input is divided into $\lceil n/N \rceil$ segments S_k, $1 \le k \le \lceil n/N \rceil$. U_k is the unpaired sequence of S_k. For each parenthesis in the input $E(j)$, $1 \le j \le n$, a function is assigned as:

$$\text{level}(j) = \begin{cases} \text{the number of \`('s to its left} \\ \quad \text{- the number of \`)' to its left} & \text{if } E(j) = \text{'('} \\ \text{the number of \`('s to its left} \\ \quad \text{- the number of \`)' to its left + 1} & \text{if } E(j) = \text{')'} \end{cases}$$

Property 3: The matching parenthesis of LMLP$_k$ in U_k is within the first unpaired sequence U_l to its right with level(LMLP$_k$) ≥ level(RMRP$_l$). The matching parenthesis of RMRP$_k$ in U_k is within the first unpaired sequence U_l to its left with level(RMRP$_k$) ≥ level(LMLP$_l$).

Proof: We prove the LMLP$_k$ case only. For RMRP$_k$, the proof is similar.

It is not possible dor the matching parenthesis of LMLP$_k$ to be within one of the unpaired sequences to its left because this matching parenthesis should be to the left of LMLP$_k$. Nor can it locate within U_k.

Suppose there exists a positive integer j, $k \le j \le l$ such that the matching right parenthesis R_k of LMLP$_k$ is within U_j. Then we have level(R_k) = level (LMLP$_k$) Because level(R_k) ≥ level (LMLP$_j$) Therefore level(LMLP$_k$) ≥ level(RMRP$_j$) This is contradictory to the fact that l is the first unpaired sequence to the right of U_k satisfying level(LMLP$_k$) ≥ level(RMRP$_j$) So, no such j exists.

Because level(LMLP$_k$) ≥ 1 and the right most right parenthesis of a proper parenthesis sequence has a level value of 1, a segment satisfying the given condition can be found for each LMLP.

The matching right parenthesis must be within U_l because level(LMLP$_k$) ≥ level(RMRP$_l$) and level(LMLP$_k$) ≤ level(RMRP$_{l-1}$) So, in the sequence of the right parentheses in U_l, there must be one parenthesis (say R_k) such that level(R_k) = level (LMLP$_k$).

From this property, an new algorithm can be found. It finds the matching parentheses for LMLPs and RMRPs by allocating the segments where the matching parentheses are allocated and then searches through the corresponding segments for the matching parentheses, This is given as Algorithm 3.

For example, let's consider a proper input of

```
( ( ( ) ) ( ( ( ) ) ( ( ) ) ) )
```

with length of 16, Suppose 4 processors are used. In step 1, input is divided into four segments and results in four unpaired sequences

```
 1            2              3             4
( (         ) ( ( (       ) ) ( (       ) ) ) )
1 2         2 2 3 4       4 3 3 4       4 3 2 1
```

with the number below as the level value assigned to each parenthesis and the number above as segment number. In step 3, the segments at which the matching parentheses of the LMLPs are allocated, are found first. Then for a similar process is applied to the RMRPs. After this step, we get

1 [4,0]	2 [4,1]	3 [4,2]	4 [0,1]
((	) (((	)) ((	))))
1 2	2 2 3 4	4 3 3 4	4 3 2 1

The two numbers within the braces are the matching segments for LMLP and RMRP respectively. In step 4, the matching parentheses for LMLP and RMRP are found and the paired parentheses are marked

# #	# # #	# #	# # #
((	) (((	)) ((	))))
1 2	2 2 3 4	4 3 3 4	4 3 2 1

Finally, step 5 pairs all the rest of the parentheses.

Algorithm 3 (SIMD-SM)

1. Partition the input into N successive segments of length n/N, and assign each segment a processor. In parallel, all the processors perform sequential pairing algorithm to find all the paired parentheses within each segment, and remove them by marking. The result is N unpaired sequences.
2. Assign each parenthesis a nesting level value by the method described by Dekel and Sahni [DeSa 83].
3. Allocate the matching unpaired sequences for all the LMLP_k, $1 \le k \le N$. In parallel, each processor k associated with LMLP_k looks for the RMRP in the next unpaired sequence U_{k+1}, then the second next U_{k+2}, and so on, until an unpaired sequence U_l is allocated which satisfies that level(LMLP_k) $\ge$ level(RMRP_l). A similar approach is also applied to RMRPs.
4. Each processor searches its unpaired sequence U_l found in step 3 for the matching right parenthesis. This is done by a linear search on U_l looking for the right parenthesis with the same level value. The paired parentheses are then marked. To avoid read conflict, a pipeline approach is used: P_1 begins first. When P_1 is searching the second parenthesis, P_2 begins, etc. A similar approach is also applied to RMRPs.
5. In parallel, each processor matches its left parentheses from the LMLP to the left until it meets a left parenthesis marked in step 4, or until all the left parentheses in this unpaired sequence are finished. A similar approach is also applied to RMRPs.

The correctness and effectiveness of algorithm 3 is given in theorem 3 below.

Theorem 3: Algorithm 3 can pair the matching parentheses for a given proper parenthesis sequence in time $t = O(n/N + N)$ with an optimal cost $c = O(n)$, for $N \le \sqrt{n}$.

Proof: The correctness of algorithm is guaranteed by the two properties of unpaired sequences. Step 1 and Step 2 are obviously correct. (See [DeSa 83] and [BaVi 85]). The correctness of Step 3 and Step 4 is given by property 3. From property 2, we can derive that step 5 is true. Regarding the time complexity and cost:

Step	1	2	3	4	5
Time	$O(n/N)$	$O(n/N)+O(\log N)$	$O(N)$	$O(n/N)$	$O(n/N)$

So, totally $t = O(n/N) + O(n/N) + O(\log N) + O(N) + O(n/N) + O(n/N)$

$$= O(n/N + N + \log N) = O(n/N + N)$$

$$c = N \times O(n/N + N)$$

$$= O(n + N^2)$$

To make algorithm 3 a cost optimal one, the condition is

$$n \ge N^2$$

So, $N \leq \sqrt{n}$

5.0 CONCLUSION

We have presented a expression parsing algorithm on SIMD-SM-R model. Its computational logic is simple, and it has a time complexity of the order of log n. Also, we proposed two optimal expression parsing algorithms on SIMD-SM model. The concurrent read simulation method can achieve a time complexity of $O(\log^2 n)$ which is comparable to Dekel and Sahni's, and has optimal cost. However, It uses additional memory units $O(Nn)$. Our second optimal algorithm doesn't need to use the additional memory units for concurrent read simulation because it avoids the use of binary tree.

Finally, a comparison of our algorithms with the other existing algorithms is given in Table 2.

Table 2. Comparisons of Algorithm Performance

	N	Model	Time	Cost=Time×N	Space
Dekel & Sahni(1983)	n	SIMD-SM	$O(\log^2 n)$	$O(n\log^2 n)$	$O(n)$
Bar-On & Vishkin(1985)	$n/\log n$	SIMD-SM-R	$O(\log n)$	$O(n)$	$O(n)$
Srikant(1990)	n	Cube-connected	$O(\log^2 n)$	$O(n\log^2 n)$	$O(n)$
Deng & Iyengar Algo.1 (1991)	n	SIMD-SM-R	$O(\log n)$	$O(n\log n)$	$O(n)$
Deng & Iyengar Algo.2 (1991)	$n/\log^2 n$	SIMD-SM	$O(\log^2 n)$	$O(n)$	$O(Nn)$
Deng & Iyengar Algo.3 (1991)	$\sqrt{n}$	SIMD-SM	$O(\sqrt{n})$	$O(n)$	$O(n)$

6.0 REFERENCE

[Akl 89] S.G.Akl, *The Design and Analysis of Parallel Algorithms*. Prentice-Hall Pr., Englewood, NJ, 1989.

[BaVi 85] I.Bar-On & U,Vishkin, Optimal Parallel Generation of a Computation Tree Forms. *ACM Trans. Program. Lang. Syst.*, 7(2). pp348-357, 1985

[DeSa 83] E.Dekel & S. Sahni, Parallel Generation of Postfix and Tree Forms. *ACM Trans. Program. Lang. Syst.*, 5(3), pp300-317, 1983.

[Knut 62] D.E.Knuth, A History of Writing Compilers, *Comput. Autom.*, pp6-19, 1962.

[MoIy 87] A.Moitra & S.S.Iyengar, *Parallel Algorithms for Some Computa-tional Problems*, 1987.

[Srik 90] Y.N. Srikant, Parallel Parsing of Arithmetic Expressions, *IEEE Trans. Comput.*, 39(1), pp130-132, 1990.

On Parametric Robust Stability and Performance

M. Dahleh*, A. Tesi** and A. Vicino***

* *Department of Mechanical Engineering, University of California*
Santa Barbara, CA 93106 (USA)

** *Dipartimento di Sistemi e Informatica, Università di Firenze*
Via di Santa Marta, 3 - 50139 Firenze (Italy)

*** *Dipartimento di Ingegneria Elettrica, Università di L'Aquila*
67040 Poggio di Roio, L'Aquila (Italy)

Abstract

In this paper we consider the problem of robust performance for systems with unstructured and parametric perturbations. The parametric uncertainty is modeled through a single-input single-output interval system description, and the unstructured uncertainty through a norm bounded perturbation. For this type of parametric robust performance we will show that in order to synthesize a controller that will achieve robust performance for a family of interval systems it is sufficient to synthesize that controller for a much smaller subset of the interval family. Regardless of the number of uncertain parameters in the original problem, this reduced set is composed of thirty two one-parameter families of systems. A special case with stronger results is also presented.

1 Introduction

In most practical control problems uncertainties are present, which represent deviations from an assumed model, and can usually take different forms. Unmodeled or difficult to model dynamics are usually represented by norm bounded perturbations. Structured perturbations are represented by structured blocks of perturbations [10,11,14], or parametric uncertainty. In any control problem the control designer has to take care of several types of perturbations, as they represent different sources of uncertainties. The problem of synthesis for robust performance deals with designing a control system that achieves robust stability and optimal disturbance rejection. We will study this problem when the uncertain system is affected by parametric and unstructured perturbations.

The aim of this paper is to bring the techniques of robust stability and synthesis closer to addressing real world problems by incorporating two types of uncertainties. We will model parametric uncertainty in terms of independent variations of the coefficients of the nominal transfer function, and unstructured uncertainty in terms of norm bounded perturbations. The techniques of μ analysis are successfully used to address the problem of robust performance when the nominal model is fixed [10,11,14]. In this paper we show that if the nominal model is a member of a single-input single-output interval family then strong extremal properties hold that help replace this potentially formidable problem by a much simpler one. Real μ-methods can be used to reformulate the current problem into a problem with block diagonal perturbations. However, the special structure in which the parameters enter the nominal model will be lost, and the extremality properties will be difficult to obtain. Upon determining the minimal subset of parameters that is sufficient for analyzing the robust performance problem, the techniques of the real μ can be used to solve the resulting problem.

This work represents a step towards a complete understanding of the extremal properties that uncertain control systems possess. Other results that share a similar goal are given in [2,6,7] for H_∞ norms, [1,4,8,9,16] for strict positive realness, [15] for frequency response analysis, and [3] for steady state analysis.

2 Problem Formulation

We will consider a class of single-input single-output interval systems perturbed by unstructured uncertainties. Let

$$\mathcal{G} = \{P = \frac{a(s)}{b(s)}, a(s) \in \mathcal{N}, b(s) \in \mathcal{D}\},$$

where $\mathcal{N}$ and $\mathcal{D}$ are the following families of real interval polynomials

$$\begin{aligned} \mathcal{N} &= \{a : a = a_0 + a_1 s + \cdots + a_p s^p : a_l \in [\alpha_l, \beta_l], l = 0, \cdots, p\}, \\ \mathcal{D} &= \{b : b = b_0 + b_1 s + \cdots + b_n s^n : b_l \in [\gamma_l, \delta_l], l = 0, \cdots, n\}. \end{aligned}$$

For a given family of interval polynomials we can define two types of special subsets: Kharitonov polynomials [12,13], and Kharitonov segments [5]. In specific for the family $\mathcal{N}$ we have the four Kharitonov polynomials

$$\begin{aligned} K_N^1(s) &= \alpha_0 + \alpha_1 s + \beta_2 s^2 + \beta_3 s^3 + \cdots, \\ K_N^2(s) &= \alpha_0 + \beta_1 s + \beta_2 s^2 + \alpha_3 s^3 + \cdots, \end{aligned}$$

$$\begin{aligned} K_N^3(s) &= \beta_0 + \alpha_1 s + \alpha_2 s^2 + \beta_3 s^3 + \cdots, \\ K_N^4(s) &= \beta_0 + \beta_1 s + \alpha_2 s^2 + \alpha_3 s^3 + \cdots, \end{aligned}$$

and the four Kharitonov segments are the following sets of convex combinations

$$\begin{aligned} S_N^1 &:= [K_N^1(s), K_N^2(s)], \\ S_N^2 &:= [K_N^1(s), K_N^3(s)], \\ S_N^3 &:= [K_N^2(s), K_N^4(s)], \\ S_N^4 &:= [K_N^3(s), K_N^4(s)]. \end{aligned}$$

For example the Kharitonov segment S_N^1 is the convex combination of K_N^1 and K_N^2 and is defined as follows

$$S_N^1(s, \lambda) = \lambda K_N^1(s) + (1 - \lambda) K_N^2(s)$$

where $\lambda \in [0, 1]$. A similar definition can be given for the Kharitonov polynomials and segments associated with the family $\mathcal{D}$. Let

$$\mathcal{G}_{\mathcal{K}} = \{ \frac{K_N^k(s)}{K_D^l(s)} \; k, l \in \{1, 2, 3, 4\} \},$$

$$\mathcal{G}_{\mathcal{KS}} = \{ \frac{S_N^k(s, \lambda)}{K_D^l(s)} \text{ or } \frac{K_N^i(s)}{S_D^j(s, \lambda)} \; k, l \in \{1, 2, 3, 4\}, \; i, j \in \{1, 2, 3, 4\}, \lambda \in [0, 1] \},$$

where S_N^k, K_N^l are the k^{th} Kharitonov segment, and the l^{th} Kharitonov polynomial associated with the family $\mathcal{N}$ and similarly for the family $\mathcal{D}$.

Define $\mathbf{\Delta}$ to be the set of all causal time-invariant operators mapping L_2 to itself, with induced L_2 norm less than or equal to one. In order to incorporate unmodeled dynamics we will consider the following families of systems

$$\Pi = \{ P = G(1 + \Delta) \; : G \in \mathcal{G}, \; \Delta \in \mathbf{\Delta} \}.$$

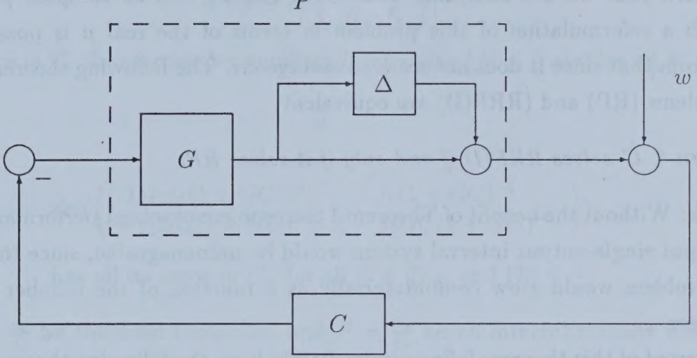
We will also need the following definitions

$$\Pi_{\mathcal{K}} = \{ P = G(1 + \Delta) \; : G \in \mathcal{G}_{\mathcal{K}}, \; \Delta \in \mathbf{\Delta} \},$$

$$\Pi_{\mathcal{KS}} = \{ P = G(1 + \Delta) \; : G \in \mathcal{G}_{\mathcal{KS}}, \; \Delta \in \mathbf{\Delta} \}.$$

The set Π is an infinite continuum of plants described parametrically, and to each point we attach a ball of unstructured perturbations. The dimension of the parametric part is dictated by the number of uncertain parameters in the description of $\mathcal{G}$. $\Pi_{\mathcal{KS}}$ is comprised of thirty two one-parameter families of interval plants, and to each plant a ball of unstructured uncertainty is attached. Note that the dimensionality of the parametric part of $\Pi_{\mathcal{KS}}$ is independent of the number of uncertain parameters in $\mathcal{G}$. Finally, in the set $\Pi_{\mathcal{K}}$ the parametric part is finite and is made up of sixteen plants.

The following configuration illustrates the problem of robust performance:



where G is a single-input single-output system in $\mathcal{G}$, Δ is an H_∞ plant perturbation with $\|\Delta\| \leq 1$, C is linear controller, and w is an L_2 disturbance. The parametric robust performance problem can be formulated as follows:

Parametric Robust Performance Problem RP: *Find a controller C such that*

1. *C stabilizes all $P \in \Pi$.*
2. *$\sup_{P\in\Pi} \|(1+PC)^{-1}\|$ is minimized.*

Consider the following two reduced problems:

Reduced Parametric Robust Performance Problem RRP(I): *Find a controller C such that*

1. *C stabilizes all $P \in \Pi_{\mathcal{K}S}$.*
2. *$\sup_{P\in\Pi_{\mathcal{K}S}} \|(1+PC)^{-1}\|$ is minimized.*

Reduced Parametric Robust Performance Problem RRP(II): *Find a controller C such that*

1. *C stabilizes all $P \in \Pi_{\mathcal{K}}$.*
2. *$\sup_{P\in\Pi_{\mathcal{K}}} \|(1+PC)^{-1}\|$ is minimized.*

3 Extremal Properties

In the case when the nominal plant is fixed and the only uncertainty present is of the unstructured type the robust performance problem is a problem in μ-synthesis. The formulation that we are analyzing deals with real as well as complex perturbations. Although a reformulation of this problem in terms of the real μ is possible, we will refrain from that since it does not seem advantageous. The following theorem shows that the problems (RP) and (RRP(I)) are equivalent

Theorem 1 *C solves RRP(I) if and only if it solves RP.*

Remark: Without the benefit of Theorem 1 the problem of robust performance for even a single-input single-output interval system would be unmanageable, since the complexity of the problem would grow combinatorially as a function of the number of uncertain parameters.

The proof of this theorem follows immediately from the following theorem (i.e. Theorem 2) which gives interesting extremal results for the μ function [10,14]. Let C be a fixed controller that stabilizes every system in $\mathcal{G}_{\mathcal{K}\mathcal{S}}$. From the generalization of Kharitonov's Theorem obtained in [5], it is clear that this controller will necessarily stabilize every system in $\mathcal{G}$. Hence, the following 2×2 interval systems are all stable

$$\mathcal{M} = \{\begin{bmatrix} (1+GC)^{-1} & (1+GC)^{-1} \\ -GC(1+GC)^{-1} & -GC(1+GC)^{-1} \end{bmatrix}, G \in \mathcal{G}\},$$

$$\mathcal{M}_{\mathcal{K}} = \{\begin{bmatrix} (1+GC)^{-1} & (1+GC)^{-1} \\ -GC(1+GC)^{-1} & -GC(1+GC)^{-1} \end{bmatrix}, G \in \mathcal{G}_{\mathcal{K}}\},$$

and

$$\mathcal{M}_{\mathcal{K}\mathcal{S}} = \{\begin{bmatrix} (1+GC)^{-1} & (1+GC)^{-1} \\ -GC(1+GC)^{-1} & -GC(1+GC)^{-1} \end{bmatrix}, G \in \mathcal{G}_{\mathcal{K}\mathcal{S}}\}.$$

Define $\mathcal{D}(n)$to be the set of all diagonal operators of the form $D = diag(\Delta_1, \ldots, \Delta_n)$ where $\Delta_i \in \mathbf{\Delta}$. Consider the perturbation structure $\mathcal{D}(2)$ of all diagonal perturbations of the form $D = Diag(\Delta_1, \Delta_2)$ where $\Delta_i \in \mathbf{\Delta}$. Let M be any element in $\mathcal{M}$, and denote by $M(j\omega)$ its evaluation at the $j\omega$ axis. Let $\bar{\sigma}(M)$ denote the maximum singular value of M. We can define [10,14] $\mu_{\mathcal{D}(2)}(M(j\omega))$ as follows

$$\frac{1}{\mu_{\mathcal{D}(2)}(M(j\omega))} := \min\{\bar{\sigma}(D) : D \in \mathcal{D}(2), \det(I + M(j\omega)D) = 0\}.$$

Theorem 2 *Let C be a single-input single-output controller that stabilizes $\mathcal{G}_{\mathcal{K}\mathcal{S}}$, then*

$$\sup_{M \in \mathcal{M}} \sup_{\omega} \mu_{\mathcal{D}(2)}(M(j\omega)) = \sup_{M \in \mathcal{M}_{\mathcal{K}\mathcal{S}}} \sup_{\omega} \mu_{\mathcal{D}(2)}(M(j\omega)).$$

Proof: We want to show that if $\sup_{M \in \mathcal{M}_{KS}} \sup_{\omega} \mu(M) < \alpha$ then $\sup_{M \in \mathcal{M}} \sup_{\omega} \mu(M) < \alpha$. Note that we are suppressing the subscripts on the μ for notational simplicity. The condition that $\sup_{M \in \mathcal{M}_{KS}} \sup_{\omega} \mu(M) < \alpha$ can be translated to the following stability condition involving interval polynomials:

$$det(I + M \begin{pmatrix} f_1 & 0 \\ 0 & f_2 \end{pmatrix})$$

has all its zeros in C_- for all complex numbers f_i satisfying $|f_i| \leq \frac{1}{\alpha}$ and for all $M \in \mathcal{M}_{KS}$.

$$\Updownarrow$$

$$det(\begin{pmatrix} 1 + f_1(1+GC)^{-1} & f_2(1+GC)^{-1} \\ -f_1 GC(1+GC)^{-1} & 1 - f_2 GC(1+GC)^{-1} \end{pmatrix})$$

has all its zeros in C_- for all $G \in \mathcal{G}_{KS}$, and $|f_i| \leq \frac{1}{\alpha}$.

Now let $C = \frac{n_c}{d_c}$ be the fixed controller, and $P = \frac{n_p}{d_p}$ be an interval transfer function in $\mathcal{G}_{KS}$. The above stability condition is equivalent to the Hurwitz stability of the following family of polynomials

$$(\delta + f_1 d_p d_c)(\delta - f_2 n_p n_c) + f_1 f_2 n_p n_c d_p d_c$$

where $\delta = n_p n_c + d_p d_c$. The previous condition reduces to the stability of

$$[n_p n_c + d_p d_c]\,[(1 - f_2) n_p n_c + (1 + f_1) d_p d_c] \tag{1}$$

for all $\frac{n_p}{d_p} \in \mathcal{G}_{KS}$. Stability of $[n_p n_c + d_p d_c]$ and $[(1 - f_2) n_p n_c + (1 + f_1) d_p d_c]$ for all $\frac{n_p}{d_p} \in \mathcal{G}_{KS}$, implies stability of $[n_p n_c + d_p d_c]$ and $[(1 - f_2) n_p n_c + (1 + f_1) d_p d_c]$ for all $\frac{n_p}{d_p} \in \mathcal{G}$, which in turn implies that $\sup_{M \in \mathcal{M}} \sup_{\omega} \mu(M) < \alpha$. ■

4 Conclusion

In this short paper we showed how the parametric robust performance problem enjoys certain extremal properties which dramatically reduce its complexity. It was also shown how the μ function possesses extreme point results in an analogous way to those for H_∞ and SPR. Using the previous results as a starting point, research is underway to further reduce the gap between parametric robustness analysis and μ-analysis.

Acknowledgement

The research of the first author is partially supported by NSF Grant ECS-8914357.

References

[1] B. D. O. Anderson, S. Dasgupta, P. Khargonekar, F. J. Kraus, amd M. Mansour, "Robust strict positive realness: characterization and construction," *IEEE Trans. on Circuits and Systems,* vol. 37, no.7,pp. 869-876, July, 1990.

[2] B. R. Barmish, and P. P. Khargonekar, " Robust Stability of Feedback Control Systems with Uncertain Parameters and Unmodeled Dynamics," *Mathematics of Control, Signals, and Systems,* vol. 3, no. 3, pp. 197-210, 1990.

[3] A. C. Bartlett, "Vertex Results for the Steady State Analysis of Uncertain Systems," *Proceedings of the 29th Conference on Decision and Control*, pp. 436-441, Hawaii, 1990.

[4] N. K. Bose and J. F. Delansky, " Boundary implications for interval positive rational functions" *IEEE Trans. on Circuit and Systems*, vol. CAS-36, pp. 454-458, 1989.

[5] H. Chapellat and S. P. Bhattacharyya, "A Generalization of Kharitonov's Theorem: Robust Stability of Interval Plants," *IEEE Trans. on Automatic Control,* vol. 34, no. 4, March 1989.

[6] H. Chapellat, M. Dahleh, and S. P. Bhattacharyya, "Robust Stability under Structured and Unstructured Perturbations," *IEEE Trans. on Automatic Control,* vol. 35, no. 10, pp. 1100-1108, October 1990.

[7] H. Chapellat, M. Dahleh, and S. P. Bhattacharyya, " Extremal Stability Manifolds in Robust Parametric Stability," *UCSB-ME-91-3, submitted for IEEE Trans. on Automatic Control.*

[8] H. Chapellat, M. Dahleh and S. P. Bhattacharyya, "On Robust Nonlinear Stability of Interval Control Systems," *IEEE Trans. on Automatic Control,* vol. 36, no. 1, January 1991.

[9] S. Dasgupta, "A Kharitonov like theorem for systems under nonlinear passive feedback," *Proceedings of the 26th Conference on Decision and Control*, pp. 2062-2063, Los Angeles, 1987.

[10] J. C. Doyle, " Analysis of Feedback Systems with Structured Uncertainties," *IEE Proceedings*, vol. 129, Part D, No. 6, 1982.

[11] M.K.H. Fan, A. L. Tits, "Characterization and Efficient Computation of the Structured Singular Value," *IEEE Trans. Automat. Contr.*, vol. AC-31, no. 8, August 1986.

[12] V. L. Kharitonov, "Asymptotic Stability of an Equilibrium Position of a Family of Systems of Linear Differential Equations," *Differntial'nye Uraveniya,* vol. 14, no. 11, pp. 1483–1485, 1987.

[13] V. L. Kharitonov, "On a Generalization of a Stability Criterion," *Akademii mauk Kahskoi SSR, fiziko-matematicheskaia,* vol. 1, pp. 53–57, 1987.

[14] A. Packard, What's new with μ: Structured uncertainty in multivariable control, Ph.D. Dissertation, UC Berkeley, 1988.

[15] A. Tesi and A. Vicino, "Kharitonov segments suffice for frequency response analysis of interval plant-controller families", *Proc. International Workshop on Robust Control*, San Antonio (USA), CRC Press, 1991.

[16] A. Vicino, and A. Tesi, "Robust strict positive realness: new results for interval plant plus controller families", *Proceedings of the 30th Conference on Decision and Control*, Brighton (UK), 1991.

Continuity properties of the real/complex structured singular value

Andy Packard & Pradeep Pandey
Mechanical Engineering
University of California
Berkeley, CA 94720

1 Introduction

The structured singular value (SSV or μ), introduced in [4],[5], is an important linear algebra tool to study a class of matrix perturbation problems. The structured singular value was introduced to take advantage of the fact that in many problems uncertainty can be represented in a structured form, e.g. a block-diagonal form. Algorithms were developed to compute upper and lower bounds for μ and the computed bounds were usually tight enough for practical applications (see [10],[6]). This led to a significant reduction in conservatism over methods which simply lump all uncertainty into a single norm-bounded block.

It is relatively straightforward to show that if the SSV is computed with respect to complex perturbations only then it is a continuous function (see Section 4). However, if real perturbations are considered, then the SSV can be discontinuous, and an example of such discontinuity is in [1]. The focus of this paper is to study these continuity properties in greater detail.

2 Definition and Properties

Our notation is standard. $\mathbf{C}^{n\times m}$ and $\mathbf{R}^{n\times m}$ are, respectively, the set of complex and real $n \times m$ matrices. For $M \in \mathbf{C}^{n\times m}$, the maximum singular value of M is denoted by $\bar{\sigma}(M)$, and M^* is the complex conjugate transpose of M. For square matrices $M \in \mathbf{C}^{n\times n}$, $\Lambda(M)$ denotes the spectrum of M, $\rho(M)$ denotes the spectral radius, and $\rho_R(M)$ denotes the *real* spectral radius. The real spectral radius is defined as

$$\rho_R(M) := \max\{|\lambda| : \lambda \in \Lambda(M) \cap \mathbf{R}\}$$

and $\rho_R(M) := 0$ if $\Lambda(M) \cap \mathbf{R} = \phi$ (the empty set). The notation $\|M\|$ will denote any matrix norm, usually the maximum singular value of the matrix.

The structured singular value, [4], is always computed with respect to a specified set $\mathbf{\Delta} \subset \mathbf{C}^{n\times n}$, usually referred to as the "block structure." This is a problem-specific set which depends on the *uncertainty* and *performance objectives* of the problem. For the moment the only assumption about $\mathbf{\Delta}$ is that it is a *closed* subset of $\mathbf{C}^{n\times n}$. Any such closed set will be called an **arbitrary block structure.**

Definition 2.1 *Consider a matrix $M \in \mathbf{C}^{n\times n}$ and a closed set $\mathbf{\Delta} \subset \mathbf{C}^{n\times n}$. The structured singular value function $\mu_{\mathbf{\Delta}} : \mathbf{C}^{n\times n} \to \mathbf{R}$ is defined via*

$$\mu_{\mathbf{\Delta}}(M) := \frac{1}{\min\{\bar{\sigma}(\Delta) : \Delta \in \mathbf{\Delta}, \det(I - M\Delta) = 0\}}, \tag{2.1}$$

unless no $\Delta \in \mathbf{\Delta}$ makes $I - M\Delta$ singular, in which case, $\mu_{\mathbf{\Delta}}(M) := 0$. ∎

Lemma 2.2 *Given a block structure $\mathbf{\Delta}$ which is closed in $\mathbf{C}^{n\times n}$, the function $\mu_{\mathbf{\Delta}} : \mathbf{C}^{n\times n} \to \mathbf{R}$ is upper semicontinuous, (upper and lower semicontinuity are defined in [12]). Equivalently, for any $M \in \mathbf{C}^{n\times n}$ and $\beta > 0$, if $\mu_{\mathbf{\Delta}}(M) < \beta$, then there is an $\alpha > 0$ such that $\mu_{\mathbf{\Delta}}(X) < \beta$ for all X with $\|X - M\| < \alpha$ (hence, if $\mu_{\mathbf{\Delta}}(M) = 0$, then $\mu_{\mathbf{\Delta}}$ is continuous at M).*

In this paper, we often make use of block partitioned matrices. Suppose $M \in \mathbf{C}^{(n_1+n_2)\times(n_1+n_2)}$ is given, and partition it in the obvious way,

$$M = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \tag{2.2}$$

with $M_{ij} \in \mathbf{C}^{n_i \times n_j}$. Let $\mathbf{\Delta}_1 \subset \mathbf{C}^{n_1 \times n_1}$ and $\mathbf{\Delta}_2 \subset \mathbf{C}^{n_2 \times n_2}$ be two block structures. Define an augmented block structure $\mathbf{\Delta}$ as

$$\mathbf{\Delta} := \{\mathrm{diag}\,[\Delta_1, \Delta_2] : \Delta_i \in \mathbf{\Delta}_i\} \tag{2.3}$$

which is compatible, in dimension, with block-partitioned matrices defined above. Also define the sets: $\mathbf{B}_i := \{\Delta_i \in \mathbf{\Delta}_i : \bar{\sigma}(\Delta_i) \leq 1\}$ and $\mathbf{B}_{\mathbf{\Delta}} := \{\mathrm{diag}\,[\Delta_1, \Delta_2] : \Delta_i \in \mathbf{B}_i\}$. Note that we can compute μ with respect to three block structures. The notation we will use to keep track of this is as follows: $\mu_1(\cdot)$ is with respect to $\mathbf{\Delta}_1$, $\mu_2(\cdot)$ is with respect to $\mathbf{\Delta}_2$, and $\mu_{\mathbf{\Delta}}(\cdot)$ is with respect to $\mathbf{\Delta}$. The inequality $\mu_{\mathbf{\Delta}}(M) \geq \max\{\mu_1(M_{11}), \mu_2(M_{22})\}$, is immediate from the definition of μ. This fact, and upper semicontinuity of μ imply the next lemma.

Lemma 2.3 *Suppose $M(\epsilon) \in \mathbf{C}^{(n_1+n_2)\times(n_1+n_2)}$ is a matrix-valued function of the scalar parameter ϵ defined by*

$$M(\epsilon) := \begin{bmatrix} M_{11} & M_{12}(\epsilon) \\ M_{21}(\epsilon) & M_{22}(\epsilon) \end{bmatrix}, \tag{2.4}$$

where $M_{ij} \in \mathbf{C}^{n_i \times n_j}$, and

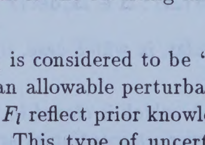
$$\lim_{\epsilon \to 0} M(\epsilon) = \begin{bmatrix} M_{11} & 0 \\ 0 & 0 \end{bmatrix}.$$

Also, suppose $\mathbf{\Delta}$ is an augmented block structure as defined in (2.3). Then $\lim_{\epsilon \to 0} \mu_{\mathbf{\Delta}}(M(\epsilon)) = \mu_1(M_{11})$.

In many control problems, it is possible to represent uncertainty as *linear fractional.* Let $\Delta_2 \in \mathbf{\Delta}_2$. If $I - M_{22}\Delta_2$ is invertible, then the **linear fractional transformation** (LFT), denoted by $F_l(M, \Delta_2)$, is **well-posed**, and defined as

$$F_l(M, \Delta_2) := M_{11} + M_{12}\Delta_2(I - M_{22}\Delta_2)^{-1}M_{21}. \tag{2.5}$$

In a feedback diagram, this appears as:



The subscript l on F_l pertains to fact that the "lower" loop of M is closed by Δ_2. An analogous formula describes $F_u(M, \Delta_1)$, which is the resulting matrix obtained by closing the "upper" loop of M with a matrix $\Delta_1 \in \mathbf{\Delta}_1$.

In this formulation the matrix M_{11} is considered to be "nominal" and $\Delta_2 \in \mathbf{B}_2$ is viewed as a norm bounded perturbation from an allowable perturbation set, $\mathbf{\Delta}_2$. The matrices M_{12}, M_{21}, and M_{22} together with the formula F_l reflect prior knowledge of how the unknown perturbation affects the nominal matrix, M_{11}. This type of uncertainty encompasses many other cases considered by researchers, including real parameter variations in differential equation models.

As the "perturbation" Δ_2 deviates from zero, the matrix $F_l(M, \Delta_2)$ deviates from M_{11}. The following theorem shows that it is possible to nonconservatively bound the quantity $\mu_1(F_l(M, \Delta_2))$ using $\mu_{\mathbf{\Delta}}(M)$.

Theorem 2.4 *For a block-partitioned matrix* $M \in \mathbf{C}^{(n_1+n_2)\times(n_1+n_2)}$ *defined in (2.2), and the uncertainty set* $\mathbf{\Delta}$ *in defined (2.3), the following equivalence holds:*

$$\mu_{\mathbf{\Delta}}(M) < 1 \quad \Leftrightarrow \quad \begin{array}{c} \mu_2(M_{22}) < 1 \\ \max\limits_{\Delta_2 \in \mathbf{B}_2} \mu_1(F_l(M, \Delta_2)) < 1 \end{array}$$

Proof: The proof follows from the definition of $\mu_{\mathbf{\Delta}}(M)$ and Schur formulas for the determinant of block partitioned matrices. Note that

$$\sup_{\Delta_2 \in \mathbf{B}_2} \mu_1(F_l(M, \Delta_2)) = \max_{\Delta_2 \in \mathbf{B}_2} \mu_1(F_l(M, \Delta_2))$$

because $\mathbf{B}_2$ is compact and μ_1 is upper semicontinuous. ∎

This theorem forms the basis for all applications of the structured singular value in linear systems theory, whether from a frequency domain, state space, or Lyapunov point of view. Note that the result holds for general block structures $\mathbf{\Delta}_1$ and $\mathbf{\Delta}_2$.

Remark 2.5 Although the structured singular value is **not necessarily a norm**, we introduce the following notation: for a proper, rational matrix P, analytic in the closed-right-half-plane, and a block structure $\mathbf{\Delta}$ of appropriate dimensions, define

$$\|P\|_{\mathbf{\Delta}} := \max\left\{ \sup_{\mathrm{Re}(s)\geq 0} \mu_{\mathbf{\Delta}}(P(s)), \mu_{\mathbf{\Delta}}(P(\infty)) \right\}.$$

3 Complex Block Structures

Strong continuity properties of the SSV can be deduced for certain special block structures which are often encountered in control problems. For example, block diagonal uncertainties were introduced in [14]. These are useful when several independent components, that together make up a larger system, are uncertain, with the uncertainty taking the form of unmodelled dynamics. Also in this setting, it is possible to put $\mathbf{H}_\infty$ norm constraints on perturbed closed-loop disturbance/error transfer functions. This framework gives rise to a special type of block structure, called a **complex block structure**.

Defining a complex block structure involves specifying three things; the type of each block, the total number of blocks, and their dimensions. We consider two types of blocks: *repeated scalar* and *full* blocks. Two nonnegative integers, S and F, represent the number of *repeated scalar* blocks and the number of *full* blocks, respectively. To keep track of their dimensions, we introduce positive integers $r_1, \ldots, r_S$; $m_1, \ldots, m_F$. The i'th repeated scalar block is $r_i \times r_i$, while the j'th full block is $m_j \times m_j$. For consistency among all the dimensions, we must have $\sum_{i=1}^{S} r_i + \sum_{j=1}^{F} m_j = n$. With those integers given, define $\mathbf{\Delta} \subset \mathbf{C}^{n\times n}$ as

$$\mathbf{\Delta} = \left\{ \mathrm{diag}\left[\delta_1 I_{r_1}, \ldots, \delta_S I_{r_S}, \Delta_1, \ldots, \Delta_F\right] : \delta_i \in \mathbf{C}, \Delta_j \in \mathbf{C}^{m_j \times m_j} \right\} \tag{3.1}$$

Associated with $\mathbf{\Delta}$ is the unit ball $\mathbf{B}_{\mathbf{\Delta}} := \{\Delta \in \mathbf{\Delta} : \bar{\sigma}(\Delta) \leq 1\}$ and the set of unitary elements $\mathbf{Q}_{\mathbf{\Delta}} := \{Q \in \mathbf{\Delta} : Q^* Q = I_n\}$.

For these complex block structures, an alternative expression for $\mu_{\mathbf{\Delta}}(M)$ follows from Definition 2.1.

Lemma 3.1 *For* $M \in \mathbf{C}^{n\times n}$ *and a complex block structure* $\mathbf{\Delta}$ *defined in (3.1),*

$$\mu_{\mathbf{\Delta}}(M) = \max_{\Delta \in \mathbf{B}_{\mathbf{\Delta}}} \rho(M\Delta).$$

Since $\mathbf{B}_{\boldsymbol{\Delta}}$ is compact, and $\rho(\cdot)$ is continuous, we have:

Corollary 3.2 *If* $\boldsymbol{\Delta}$ *is a complex block structure as defined in (3.1), then* $\mu_{\boldsymbol{\Delta}}$ *is continuous.*

An result from [4] concerning complex block structures is that the maximum in Lemma 3.1 is achieved on the unitary elements of $\boldsymbol{\Delta}$. The proof of this uses Rouche's Theorem, [12].

Lemma 3.3 *For* $M \in \mathbf{C}^{n \times n}$ *and a complex block structure* $\boldsymbol{\Delta}$ *defined in (3.1),*

$$\mu_{\boldsymbol{\Delta}}(M) = \max_{Q \in \mathbf{Q}_{\boldsymbol{\Delta}}} \rho(MQ).$$

This fact is important since it reduces some of the computational burden in computing lower bounds for μ. Also, this has implications on subharmonicity and maximum modulus results which are studied in [2]. In [9], Lemma 3.3 was extended to linear fractional transformations in case both $\boldsymbol{\Delta}_1$ and $\boldsymbol{\Delta}_2$, and consequently $\boldsymbol{\Delta}$, are **complex block structures.**

Lemma 3.4 *Let* $M \in \mathbf{C}^{(n_1+n_2)\times(n_1+n_2)}$ *be a block-partitioned matrix, as defined in (2.2). Also, let* $\boldsymbol{\Delta}_1 \subset \mathbf{C}^{(n_1)\times(n_1)}$ *and* $\boldsymbol{\Delta}_2 \subset \mathbf{C}^{(n_2)\times(n_2)}$ *be complex block structures as in (3.1). If* $\mu_2(M_{22}) < 1$, *then*

$$\max_{Q_2 \in \mathbf{Q}_2} \mu_1(F_l(M, Q_2)) = \max_{\Delta_2 \in \mathbf{B}_2} \mu_1(F_l(M, \Delta_2)).$$

We shall see in Section 6 that this lemma simplifies certain robustness tests.

4 General Continuity Lemmas

Consider matrices partitioned as in (2.2), and augmented block structures defined in (2.3). Assume that $\boldsymbol{\Delta}_2$ is a **complex** block structure, as defined in (3.1).

Lemma 4.1 *Consider* M *and* $\boldsymbol{\Delta}$ *defined in (2.2)–(2.3) where* $\boldsymbol{\Delta}_2 \subset \mathbf{C}^{n_2 \times n_2}$ *is a complex block structure and* $\boldsymbol{\Delta}_1$ *is any closed subset of* $\mathbf{C}^{n_1 \times n_1}$. *If* $\mu_1(M_{11}) < \mu_{\boldsymbol{\Delta}}(M) =: \beta$ *then* $\mu_{\boldsymbol{\Delta}}$ *is lower semicontinuous at* M. *Equivalently, for any* $\epsilon > 0$ *there is an* $\alpha > 0$ *such that* $\mu_{\boldsymbol{\Delta}}(X) > \beta - \epsilon$ *for all* X *such that* $\|X - M\| < \alpha$.

Using the above notation, several corollaries are immediate consequences of Lemma 2.2 and Lemma 4.1.

Corollary 4.2 *If* $\mu_{\boldsymbol{\Delta}}(M) > \mu_1(M_{11})$, *then* $\mu_{\boldsymbol{\Delta}}$ *is continuous at* M.

Corollary 4.3 *If* $\mu_{\boldsymbol{\Delta}}(M) > \mu_1(M_{11})$, *there exists an* $\epsilon > 0$ *such that if* $X \in \mathbf{C}^{(n_1+n_2)\times(n_1+n_2)}$, *and* $\|X - M\| < \epsilon$, *then* $\mu_{\boldsymbol{\Delta}}(X) > \mu_1(X_{11})$.

Corollary 4.4 *If* $\mu_{\boldsymbol{\Delta}}(M) > \mu_1(M_{11})$, *there exists an* $\epsilon > 0$ *such that* $\mu_{\boldsymbol{\Delta}}$ *is continuous on the set*

$$\left\{X \in \mathbf{C}^{(n_1+n_2)\times(n_1+n_2)} : \|X - M\| < \epsilon\right\}.$$

Now we consider situations that exploit Lemma 4.1 via the condition $\mu_{\boldsymbol{\Delta}}(M) > \mu_1(M_{11})$. In the first case this condition is guaranteed by imposing certain rank conditions on the blocks of the matrix M.

Lemma 4.5 *Consider M and $\mathbf{\Delta}$ defined in (2.2)–(2.3). Suppose that $\mathbf{\Delta}_2 = \mathbf{C}^{n_2 \times n_2}$ (i.e., a single full complex block), and that M_{21} and M_{12} are, respectively, full column rank and full row rank (this implicitly requires $n_2 \geq n_1$). Then $\mu_{\mathbf{\Delta}}(M) > \mu_1(M_{11})$ and, consequently, $\mu_{\mathbf{\Delta}}$ is continuous at M.*

Another way to use the result in Lemma 4.1 is as follows. Let $P \in \mathbf{C}^{n \times n}$ be given, along with invertible matrices $W_L, W_R \in \mathbf{\Delta}_2$. Define $M \in \mathbf{C}^{2n \times 2n}$ as

$$M := \begin{bmatrix} P & PW_R \\ W_L P & W_L P W_R \end{bmatrix}. \tag{4.1}$$

Lemma 4.6 *Consider $M \in \mathbf{C}^{2n \times 2n}$ defined in (4.1). Suppose $\mathbf{\Delta}_2 \subset \mathbf{C}^{n \times n}$ is a complex block structure and that $\mathbf{\Delta}_1 := \mathbf{R}^{n \times n} \bigcap \mathbf{\Delta}_2$. If $\mu_2(P) > 0$, then $\mu_1(P) < \mu_{\mathbf{\Delta}}(M)$ and $\mu_{\mathbf{\Delta}}$ is continuous at M.*

As a special case of the previous lemma, consider $W_L = W_R = \sqrt{\epsilon} I$.

Lemma 4.7 *Define*

$$M(\epsilon, P) := \begin{bmatrix} P & \sqrt{\epsilon} P \\ \sqrt{\epsilon} P & \epsilon P \end{bmatrix},$$

and the function f as

$$f(\epsilon, P) := \mu_{\mathbf{\Delta}}(M(\epsilon, P)).$$

Then, the function $f : (0, \infty) \times \mathbf{C}^{n \times n} \to \mathbf{R}$ is continuous, $f(\epsilon, P) \geq \mu_1(P)$, and

$$\lim_{\epsilon \to 0} f(\epsilon, P) = \mu_1(P).$$

A similar result is obtained by augmenting the real uncertainty with imaginary uncertainty.

Lemma 4.8 *Assume that $\mathbf{\Delta}_1 = \mathbf{R}^{n \times n} \bigcap \mathbf{\Delta}_C$ for some complex block structure $\mathbf{\Delta}_C$, and let $\mathbf{\Delta}_2 := \mathbf{\Delta}_1$. Define*

$$M(\epsilon, P) := \begin{bmatrix} P & \sqrt{\epsilon} P \\ j\sqrt{\epsilon} P & j\epsilon P \end{bmatrix},$$

$\mathbf{\Delta} := \{\text{diag}[\Delta_1, \Delta_2] : \Delta_i \in \mathbf{\Delta}_i\}$, and

$$f(\epsilon, P) := \mu_{\mathbf{\Delta}}(M(\epsilon, P)).$$

Then, the function $f : (0, \infty) \times \mathbf{C}^{n \times n} \to \mathbf{R}$ is continuous, $f(\epsilon, P) \geq \mu_1(P)$, and

$$\lim_{\epsilon \to 0} f(\epsilon, P) = \mu_1(P).$$

5 Real/Complex μ Problems

In the previous sections, we have defined **arbitrary** and **complex** block structures. A closed subset $\mathbf{\Delta} \subset \mathbf{R}^{n \times n}$ is called a **real block structure** if for all $\alpha \in \mathbf{R}$, $\alpha \neq 0$,

$$\mathbf{\Delta} = \{\alpha \Delta : \Delta \in \mathbf{\Delta}\} =: \alpha \mathbf{\Delta}. \tag{5.1}$$

Suppose $\mathbf{\Delta}_1 \subset \mathbf{R}^{n_1 \times n_1}$ is a real block structure, and define $\mathbf{B}_1$ and $\mathbf{Q}_1$ as follows:

$$\mathbf{B}_1 = \mathbf{Q}_1 := \{\Delta_1 \in \mathbf{\Delta}_1 : \bar{\sigma}(\Delta_1) \leq 1\}.$$

Let $\mathbf{\Delta}_2 \subset \mathbf{C}^{n_2 \times n_2}$ be a complex block structure, with associated sets $\mathbf{B}_2$ and $\mathbf{Q}_2$. Now $\mathbf{\Delta}$, $\mathbf{B}_{\mathbf{\Delta}}$, and $\mathbf{Q}_{\mathbf{\Delta}}$ are the obvious diagonal augmentations involving $\mathbf{\Delta}_1$, $\mathbf{\Delta}_2$, $\mathbf{B}_1$, $\mathbf{B}_2$, $\mathbf{Q}_1$, and $\mathbf{Q}_2$. Lemma 3.3, which provides an alternative characterization of $\mu_{\mathbf{\Delta}}(M)$ for complex block structures, was extended in [15] for such mixed block structures. As stated below, the next theorem is a slight extension of Theorem 1 in [15]. However, the proof is nearly identical and is omitted.

Lemma 5.1 *Consider an augmented block structure* $\mathbf{\Delta}$ *where* $\mathbf{\Delta}_1$ *is a real block structure defined in (5.1) and* $\mathbf{\Delta}_2$ *is a complex block structure. Then for* $M \in \mathbf{C}^{n \times n}$

$$\mu_{\mathbf{\Delta}}(M) = \max_{Q \in \mathbf{Q}_{\mathbf{\Delta}}} \rho_R(MQ).$$

Similarly, Lemma 3.4 can be extended to include real block structures.

Lemma 5.2 *Let* $M \in \mathbf{C}^{(n_1+n_2)\times(n_1+n_2)}$ *be a block-partitioned matrix, as defined in (2.2), Let* $\mathbf{\Delta}_1 \subset \mathbf{C}^{n_1 \times n_1}$ *be a real block structure as defined in (5.1) and* $\mathbf{\Delta}_2 \subset \mathbf{C}^{n_2 \times n_2}$ *be complex block structure. Suppose* $\mu_2(M_{22}) < 1$. *Then*

$$\max_{\Delta_2 \in \mathbf{B}_2} \mu_1 (F_l(M, \Delta_2)) = \max_{Q_2 \in \mathbf{Q}_2} \mu_1 (F_l(M, Q_2)).$$

Remark 5.3 In this Lemma, the set $\mathbf{\Delta}_1$ can actually be **any** diagonal augmentation of real and complex block structures.

This lemma implies that certain robustness conditions can be verified by examining the unit circle rather than the unit disc. For example, suppose $\rho(A) < 1$ and define $G(z) := D + C(zI - A)^{-1}B$ for $z \in \mathbf{C}$. Define

$$M := \begin{bmatrix} D & C \\ B & A \end{bmatrix}.$$

Then $F_l(M, \frac{1}{z}I_n) = G(z)$. Because of Lemma 5.2, we can write

$$\sup_{|z| \geq 1} \mu_1 (G(z)) = \max_{|z|=1} \mu_1 (G(z)). \tag{5.2}$$

These results can be applied to continuous-time systems by the bilinear transform equivalence of the open left-half-plane and the open unit disc. However, extrapolating these ideas to right-half-plane/imaginary axis results must be done with care. Consider the following example. Suppose that G is a real-rational function, analytic in the closed right-half plane. In particular, let

$$G(s) := \frac{2s+1}{s+1}.$$

Let $\mathbf{\Delta} := \mathbf{R}$. Note that

$$\begin{aligned} \sup_{\omega \in \mathbf{R}} \mu_{\mathbf{\Delta}}(G(j\omega)) &= 1, \\ \sup_{\mathrm{Re}(s) \geq 0} \mu_{\mathbf{\Delta}}(G(s)) &= 2, \\ \mu_{\mathbf{\Delta}}\left(\lim_{s \to \infty} G(s)\right) &= 2. \end{aligned}$$

Hence, the supremum over the imaginary axis is not equal to the supremum over the right-half plane. This is due to the discontinuity at $s = \infty$. This problem does not arise in the discrete-time system. Indeed, in discrete time, i.e., for $s = \frac{1}{2}\frac{z-1}{z+1}$ we have

$$G(z) = \frac{4z}{3z+1}.$$

The only values of z, with $|z| = 1$, for which $\mu_1(G(z))$ is nonzero are $z = 1$ and $z = -1$, and

$$\max_{|z|=1} \mu_1 (G(z)) = 2.$$

The following theorem is the analog of (5.2) for continuous-time systems.

Theorem 5.4 *Suppose $P(s)$ is a $m \times m$ real-rational, proper transfer function matrix, which is analytic in the closed right-half-plane. Define $P_\infty := \lim_{s\to\infty} P(s)$. Let $\mathbf{\Delta}$ be any diagonal augmentation of the real and complex block structures defined in (5.1) and (3.1). Then*

$$\max\left\{\sup_{\mathrm{Re}(s)\geq 0}\mu_{\mathbf{\Delta}}(P(s))\,,\ \mu_{\mathbf{\Delta}}(P_\infty)\right\} = \max\left\{\sup_{\omega\in\mathbf{R}}\mu_{\mathbf{\Delta}}(P(j\omega))\,,\ \mu_{\mathbf{\Delta}}(P_\infty)\right\}.$$

Finally, for a certain class of real block structures, the function $\mu_{\mathbf{\Delta}}(\cdot)$ is continuous at all real matrices, M.

Lemma 5.5 *Let $\mathbf{\Delta}$ be a real block structure defined as*

$$\mathbf{\Delta} := \{\Delta \in \mathbf{R}^{n\times n} : \Delta = \mathrm{diag}[\delta_1, \ldots, \delta_n],\ \delta_i \in \mathbf{R}\}. \tag{5.3}$$

Then, for any $M \in \mathbf{R}^{n\times n}$, $\mu_{\mathbf{\Delta}} : \mathbf{R}^{n\times n} \to \mathbf{R}$ is continuous at M.

Remark 5.6 The above result readily extends to the case where $\mathbf{\Delta}$ has full real blocks in addition to scalar reals, as depicted below.

$$\mathbf{\Delta} := \{\Delta : \Delta = \mathrm{diag}[\delta_1, \ldots, \delta_r, \Delta_1, \ldots, \Delta_k],\ \delta_i \in \mathbf{R}, \Delta_i \in \mathbf{R}^{n_i\times n_i}\} \tag{5.4}$$

Remark 5.7 The above result is **not** true if $\mathbf{\Delta}$ contains repeated scalar blocks. For example, suppose $\mathbf{\Delta} := \{\delta I : \delta \in \mathbf{R}\}$, so that $\mu_{\mathbf{\Delta}}(M) = \rho_R(M)$. This quantity need not vary continuously with M. In particular, let

$$M_\epsilon := \begin{bmatrix} 1 & \epsilon \\ -\epsilon & 1 \end{bmatrix}.$$

For $\epsilon = 0$, we have $\mu_{\mathbf{\Delta}}(M_\epsilon) = 1$. However, for any $\epsilon > 0$, $\mu_{\mathbf{\Delta}}(M_\epsilon) = 0$.

6 Robustness Margins

Consider a linear system with linear fractional uncertainty, as shown below:

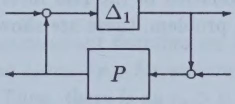


Figure 6.1

Here $P(s)$ is a stable, proper, rational, $n \times n$ transfer function and the linear fractional perturbation is the Δ_1 element, which lies in some prespecified block structure $\mathbf{\Delta}_1 \subset \mathbf{C}^{n\times n}$. Suppose that $\mathbf{\Delta}_2$ is a **complex** block structure and $\mathbf{\Delta}_1 = \mathbf{R}^{n\times n} \bigcap \mathbf{\Delta}_2$. Let $\mathbf{\Delta}$ be the diagonal augmentation of these two structures.

Based on results obtained in the previous sections, consider two related problems:

- **Problem 1:** μ calculation for Real uncertainty,

$$\|P\|_{\mathbf{\Delta}_1} = \max\left\{\sup_{\omega\in\mathbf{R}}\mu_1(P(j\omega)), \mu_1(P(\infty))\right\} =: \frac{1}{R_1}.$$

  For any $\beta > 0$, the closed-loop interconnection in Figure 6.1 is well-posed and stable for all $\Delta_1 \in \mathbf{\Delta}_1$ with $\bar{\sigma}(\Delta_1) \leq \beta$ if and only if $\beta < R_1$.

- **Problem 2:** μ calculation for Real+Complex uncertainty. For $\epsilon > 0$ define $M(s)$ as

$$M(s) := \begin{bmatrix} P(s) & \sqrt{\epsilon}P(s) \\ \sqrt{\epsilon}P(s) & \epsilon P(s) \end{bmatrix}$$

and the μ calculation for Real+Complex uncertainty,

$$\|M\|_{\boldsymbol{\Delta}} = \max\left\{ \sup_{\omega\in\mathbf{R}} \mu_{\boldsymbol{\Delta}}\left(M(j\omega)\right), \mu_{\boldsymbol{\Delta}}\left(M(\infty)\right) \right\} =: \frac{1}{R_2}.$$

The interconnection associated with Problem 2 is shown below.

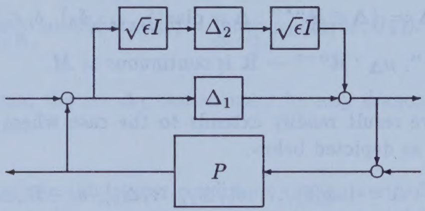
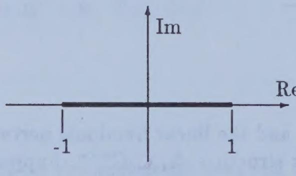


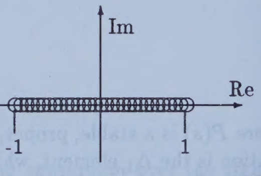
Figure 6.2

Note that this is similar to the interconnection associated with Problem 1. However, now the linear fractional perturbation to P is $\Delta_1 + \epsilon\Delta_2$, where Δ_2 has the same structure as Δ_1, but may take on complex values. The value R_2 has the interpretation that for any $\beta > 0$, this closed-loop interconnection is well-posed and stable for all $\Delta_i \in \boldsymbol{\Delta}_i$ with $\bar{\sigma}(\Delta_i) \leq \beta$ if and only if $\beta < R_2$.

Note that Problem 1 involves purely real uncertainty, while the Problem 2 pertains to "nearly real" uncertainty. In the case of a scalar uncertainty block, it is easy to give a pictorial representation of the "unit" perturbations involved. In Problem 1, it is the interval $[-1, 1]$, while in Problem 2 it is the setwise summation of the real interval, and the complex disc of radius ϵ, centered at the origin. For each problem, these are shown below.



Problem 1, Unit Ball Uncertainty



Problem 2, "Unit Ball" Uncertainty

For these two Problems, Theorem 6.1 (upcoming) confirms the following facts:

1. The margin R_2 computed in Problem 2 will be a continuous function of the problem data (ie. $P(s)$) (note that this has nothing to do with the metod of computation).
2. In Problem 2, the structured singular value will be continuous, so that the frequency domain maximization calculation of $\|M\|_{\boldsymbol{\Delta}}$ via gridding is possible (as the grid gets finer, the calculated maximum approaches the actual maximum).

3. Under the assumption of exact computation, and exact numerical representation, the margin R_2 approaches the margin R_1 as ϵ gets arbitrarily small.

4. If R_2 is significantly less than R_1, then **from an engineering point of view** the uncertainty model probably needs more attention. This is because in most engineering models, certain assumptions are made which ignore the dynamics of individual components. Without additional study though, the Problem 2 margin will always be on the conservative side of the Problem 1 margin.

Theorem 6.1 *Suppose $P(s)$ is a stable, proper, rational $n \times n$ transfer matrix. Define the set $\mathcal{P}$ as follows*

$$\mathcal{P} := \{P(j\omega) : \omega \in [0, \infty)\} \cup \{P_\infty\}. \tag{6.1}$$

Suppose that $\mathbf{\Delta}_2 \subset \mathbf{C}^{n\times n}$ is a complex block structure, and the real block structure $\mathbf{\Delta}_1 \subset \mathbf{R}^{n\times n}$ is related to $\mathbf{\Delta}_2$ by

$$\mathbf{\Delta}_1 = \mathbf{R}^{n\times n} \bigcap \mathbf{\Delta}_2.$$

Let $\mathbf{\Delta}$ be the diagonal augmentation of these two sets,

$$\mathbf{\Delta} := \{\mathrm{diag}\,[\Delta_1, \Delta_2] : \Delta_1 \in \mathbf{\Delta}_1, \Delta_2 \in \mathbf{\Delta}_2\}.$$

For any $P \in \mathcal{P}$ we define

$$M(\epsilon, P) := \begin{bmatrix} P & \epsilon P \\ \epsilon P & \epsilon^2 P \end{bmatrix},$$

and the function

$$f(\epsilon, P) := \mu_{\mathbf{\Delta}}(M(\epsilon, P)).$$

The function $f : (0, \infty) \times \mathbf{C}^{n\times n} \to \mathbf{R}$ is continuous. For $\epsilon > 0$, we have $f(\epsilon, P) \geq \mu_1(P)$ and

$$\lim_{\epsilon \to 0} f(\epsilon, P) = \mu_1(P).$$

Further, given any $\delta > 0$ there is an $\epsilon_* > 0$ such that

$$\|M(\epsilon^*, P)\|_{\mathbf{\Delta}} = \max_{P\in\mathcal{P}} f(\epsilon_*, P) < \|P\|_{\mathbf{\Delta}_1} + \delta.$$

Corollary 6.2 *Let $g(\omega)$ be a piecewise-constant function on $\mathbf{R}$ that takes on a finite set of values. As notation, let g_∞ be the value taken by g for arbitrarily large values of ω. Suppose that for all finite ω, $\mu_1[P(\omega)] < g(\omega)$. Then, there is an $\epsilon_* > 0$ such that for all $\epsilon \in (0, \epsilon_*)$ and all ω,*

$$\begin{array}{rcccl} \mu_1[P(\omega)] & \leq & \mu_{\mathbf{\Delta}}(M_\epsilon(\omega)) & \leq & g(\omega) \\ \mu_1[P_\infty] & \leq & \mu_{\mathbf{\Delta}}(M_\epsilon(\infty)) & \leq & g_\infty. \end{array}$$

In closing, we note that the same four properties which relate R_2 to R_1 can be derived for the regularization methods given in Lemmas 4.5 and 4.8.

7 Acknowledgements

The authors gratefully acknowledge helpful advice from J. Doyle, P. Khargonekar, S. Shahruz, T. Sideris, and A. Tits and financial support from the National Science Foundation, award ECS-8909499.

References

[1] B. Barmish, P. Khargonekar, Z. Shi, and R. Tempo, "Robustness margin need not be a continuous function of the problem data," *System and Control Letters*, vol. 15, pp. 91–98, 1989.

[2] S. Boyd and C.A. Desoer, "Subharmonic functions and performance bounds on linear time-invariant feedback systems," *IMA J. of Mathematical Control and Information*, volume 2, pp. 153-170, 1985.

[3] R. deGaston and M. Safonov, "Exact calculation of the multiloop stability margin," *IEEE Transactions on Automatic Control*, vol. AC-33, pp. 156–171, Feb. 1988.

[4] J.C. Doyle, "Analysis of feedback systems with structured uncertainties," *IEE Proceedings*, Part D, vol. 129, pp. 242–250, 1982.

[5] J.C. Doyle, J.E. Wall, and G. Stein, "Performance and robustness analysis for structured uncertainty," *Proceedings of the 21th Conference on Decision and Control*, pp. 629–636, 1982.

[6] M.K.H. Fan and A.L. Tits, "Characterization and efficient computation of the structured singular value," *IEEE Transactions on Automatic Control*, vol. AC-31, pp. 734–743, Aug. 1986.

[7] M.K.H. Fan, A.L. Tits, and J.C. Doyle, "Robustness in the presence of mixed parametric uncertainty and unmodelled dynamics," *IEEE Transactions on Automatic Control*, vol. AC-36, pp. 25–38, January 1991.

[8] P. Khargonekar and I. Kaminer, "Robust stability analysis with structured, norm-bounded, unstable uncertainty," *1991 American Control Conference Proceedings*, pp. 2700-2701.

[9] A. Packard and J.W. Balsamo, "A maximum modulus theorem for linear fractional transformations," *System and Control Letters*, vol. 11, pp. 365–367, 1989.

[10] A. Packard, M. Fan, and J. Doyle, "A power method for the structured singular value," *28th IEEE CDC Proceedings*, pp. 2132-2137, 1988.

[11] Y.K. Foo and I. Poslethwaite, " Extensions of the small-μ test for robust stability ," *IEEE Transactions on Automatic Control*, vol. AC-33, pp. 172–176, Feb., 1988.

[12] W. Rudin, "Real and Complex Analysis," McGraw Hill Book Company, New York, 1966.

[13] A. Sideris and R. Sanchez Pena, "Robustness margin calculation with dynamic and real parametric uncertainty," *IEEE Transactions on Automatic Control*, vol. AC-35, pp. 970–974, Aug., 1990.

[14] M.G. Safonov and M. Athens, *IEEE Transactions on Automatic Control*, "Gain and phase margin for multiloop LQG regulators," vol. AC-22, pp. 173-179, April, 1977.

[15] P.M. Young and J.C. Doyle, "Computation of μ with real and complex uncertainties," *Proceedings of the 29th Conference on Decision and Control*, pp. 1230–1235, 1990.

Systems with Structured Uncertainty: Relations between Quadratic and Robust Stability *

M. A. Rotea, M. Corless, D. Da
School of Aeronautics and Astronautics
Purdue University, West Lafayette, IN 47907
and
I. R. Petersen
Department of Electrical Engineering
Australian Defence Force Academy, Campbell, ACT 2800

Abstract

The purpose of this note is to investigate the relation between the notions of *robust stability* and *quadratic stability* for uncertain systems with structured uncertainty due to both real and complex parameter variations. We present examples which demonstrate that for systems containing at least two uncertain blocks, the notions of robust stability for complex parameter variations and quadratic stability for real parameter variations are not equivalent; in fact neither implies the other. A byproduct of these examples is that, for this class of systems, quadratic stability for real perturbations need not imply quadratic stability for complex perturbations. This is in stark contrast with the situation in the case of unstructured uncertainty, for which it is known that quadratic stability for either real or complex perturbations is equivalent to robust stability for complex perturbations, and thus equivalent to a small gain condition on the transfer matrix that the perturbation experiences.

1 Introduction

In this note we study the relation between the notions of *robust* and *quadratic* stability. The focus is on uncertain systems with so-called "norm bounded structured uncertainty," due to both real and complex parameter variations. Robust stability means that the uncertain system is stable for all (constant but otherwise unknown) parameter variations, while quadratic stability means that the uncertain system admits a parameter independent (quadratic) Lyapunov function which guarantees stability for all memoryless, possibly time-varying and/or state-dependent, parameter variations. Precise definitions are given later in Section 2.

By definition, quadratic stability is a stronger requirement than robust stability. Similarly, robust/quadratic stability with respect to complex parameter variations is stronger

---

*This work was supported in part by the National Science Foundation under Grants no. ECS-91-08493 and MSM-90-57079.

than robust/quadratic stability with respect to real parameter variations. A surprising result is that, in the case of *one block of uncertainty*, the notions of quadratic stability for real parameter variations, quadratic stability for complex parameter variations, and robust stability for complex parameter variations, are all equivalent. See, for example, Khargonekar et al. [4].

In this paper we show that the result of Khargonekar et al. does not carry over to systems containing at least *two blocks of uncertainty.* More specifically, we will present examples that confirm the following natural expectations:

i) There are uncertain systems which are stable for all *complex* (constant but otherwise unknown) parameter variations, but not quadratically stable for all *real* (possibly time-varying/state-dependent) memoryless parameter variations.

ii) There are uncertain systems which are quadratically stable for all *real* (possibly time-varying/state-dependent) memoryless parameter variations, but not stable for all *complex* (constant but otherwise unknown) parameter variations.

Similar conclusions were previously obtained by Packard and Doyle [6] in a discrete-time context. Our examples differ from those of Packard and Doyle in two ways. Firstly, we work with continuous-time systems. More importantly, the examples provided in this work are "linear affine" in the uncertain parameters. Since these examples are of a "negative" nature, our results are slightly stronger than those of Packard and Doyle.

Our notation is fairly standard. The real part and maximum singular value of a complex matrix are denoted by $\Re(\cdot)$ and $\overline{\sigma}(\cdot)$, respectively. Given a matrix-valued function G (of a complex variable) which is bounded on the imaginary axis, we let $\|G\|_\infty := \sup_{\omega \in R} \overline{\sigma}(G(j\omega))$.

2 Stability notions for uncertain systems

Consider the uncertain system

$$\dot{x}(t) = F(\Delta(t, x(t)))x(t), \tag{1}$$

where $t \in \boldsymbol{R}$ is the time variable, $x(t) \in \boldsymbol{E}^n$ is the state-vector, $F(\cdot)$ is a known matrix-valued function, and $\Delta(\cdot)$ represents *parameter uncertainty* which is possibly time-varying and/or state dependent. Here, $\boldsymbol{E}$ stands for the field over which the state-vector and the uncertain parameters are defined. In this paper $\boldsymbol{E}$ is either the set of real ($\boldsymbol{R}$) or complex ($\boldsymbol{C}$) numbers. The bounding set for the uncertain parameters will be denoted by $\boldsymbol{\Delta}_E$, and we assume that $\boldsymbol{\Delta}_E$ is a known compact subset of $\boldsymbol{E}^{m \times p}$. We also assume that the mapping $\Delta(\cdot) : \boldsymbol{R} \times \boldsymbol{E}^n \to \boldsymbol{\Delta}_E$ is Lebesgue measurable in the first argument and continuous in the second argument; we shall call such a function an *admissible uncertainty*. Finally, we also assume that the matrix-valued function $F(\cdot)$ is a continuous mapping from $\boldsymbol{\Delta}_E$ into $\boldsymbol{E}^{n \times n}$.

When the uncertain parameters are time-varying and/or they depend on the state-vector, a popular stability notion for the uncertain system (1) is the following notion of *quadratic stability.*

Definition 2.1 *The uncertain system (1) is* quadratically stable *with respect to the bounding set* $\boldsymbol{\Delta}_E$ *if there exists a positive definite hermitian matrix* $P \in \boldsymbol{E}^{n \times n}$ *such that, for any nonzero* $x \in \boldsymbol{E}^n$ *and* $\Delta \in \boldsymbol{\Delta}_E$, *the following condition is satisfied*

$$\phi(x, \Delta) := \Re(x^* P F(\Delta) x) < 0. \tag{2}$$

Note that *there is no distinction between time-varying/state-dependent and constant perturbations as far as the notion of quadratic stability is concerned.* That is, if there exists a positive definite hermitian matrix P such that (2) holds, the Lyapunov function given by $V(x) = x^*Px$ ensures global uniform asymptotic stability of (1) not only for any constant perturbation $\Delta \in \mathbf{\Delta}_E$ but also for any "admissible uncertainty" $\Delta(\cdot)$.

The notion of quadratic stability has proven to be quite useful for developing "systematic" synthesis procedures for robust controller design. These synthesis results are especially useful when the uncertain parameters are time-varying and/or state dependent. See, for example, the compilation of papers by Dorato [1], and Dorato and Yedavalli [2], for some recent developments in this area.

When the uncertain parameters are constant (but otherwise unknown) elements of the bounding set, the notion of quadratic stability becomes a fairly strong requirement; in this case it is more natural to call for the less restrictive requirement of *robust stability.*

Definition 2.2 *The uncertain system (1) is* robustly stable *with respect to the bounding set $\mathbf{\Delta}_E$ if for any $\Delta \in \mathbf{\Delta}_E$, $F(\Delta)$ has all eigenvalues in the open left half complex plane.*

In the sequel, we will say that the uncertain system (1) is $\mathrm{RS}(\mathbf{\Delta}_E)$ (resp., $\mathrm{QS}(\mathbf{\Delta}_E)$) when (1) is robustly stable (resp., quadratically stable) with respect to the bounding set $\mathbf{\Delta}_E$.

Consider the uncertain system (1). In this note we are primarily interested in the case in which the function $F(\cdot)$ and the bounding set $\mathbf{\Delta}_E$ are given by:

$$F(\Delta) := A + B\Delta C \tag{3}$$

$$\mathbf{\Delta}_E := \{\Delta = \mathrm{block\_diag}(\Delta_1, \ldots, \Delta_r) : \forall i = 1, \ldots, r, \Delta_i \in \boldsymbol{E}^{m_i \times p_i}, \overline{\sigma}(\Delta_i) \leq 1\}, \tag{4}$$

where A, B, C are known real matrices of compatible dimensions. This type of uncertainty modeling encompasses many cases of practical significance. Note that $F(\Delta)$ can also be written as

$$F(\Delta) := A + \sum_{i=1}^{r} B_i \Delta_i C_i,$$

where

$$B = [B_1 B_2 \ldots B_r], \qquad C = [C_1' C_2' \ldots C_r']'.$$

We shall call an uncertain system of the form (1) that satisfies (3) and (4) a system with *norm bounded uncertainty.* Although the set $\mathbf{\Delta}_E$ introduced in (4) depends on the number of uncertain blocks and their dimensions, we shall not carry this dependence explicitly to avoid cumbersome notation.

An important result in the analysis of uncertain systems with norm bounded uncertainty is the following:

Theorem 2.1 ([4]) *Let Σ denote a system with norm bounded uncertainty as defined in (1) and (3). Define $G(s) := C(sI - A)^{-1}B$. Let $\boldsymbol{E}$ denote either $\boldsymbol{R}$ or $\boldsymbol{C}$. Suppose that the bounding set $\mathbf{\Delta}_E$ is given by*

$$\mathbf{\Delta}_E := \{\Delta \in \boldsymbol{E}^{m \times p} : \overline{\sigma}(\Delta) \leq 1\}.$$

Then, the following statements are equivalent:

1. *Σ is* $\mathrm{QS}(\mathbf{\Delta}_C)$.
2. *Σ is* $\mathrm{QS}(\mathbf{\Delta}_R)$.

3. Σ is RS$(\mathbf{\Delta}_C)$.

4. The matrix A has all eigenvalues in the open left half complex plane and $\|G\|_\infty < 1$.

The contribution of this note is to show that for uncertain systems with norm bounded "structured" uncertainty, i.e. two or more uncertain blocks in (4), statements 1-3 of Theorem 2.1 are no longer equivalent. Indeed, we give examples of systems with norm bounded structured uncertainty which demonstrate that regions 2 and 4 in the Venn diagram of Figure 1 are nonempty. The combination of these examples together with the well-known fact that there are systems with norm bounded uncertainty (see [3] for an example with just one block of uncertainty) that are RS$(\mathbf{\Delta}_R)$ but neither RS$(\mathbf{\Delta}_C)$ nor QS$(\mathbf{\Delta}_R)$, i.e. region 1 is nonempty, permit us to conclude that, in general, no two of the above mentioned stability notions are equivalent. Even though we have not been able to produce an example that demonstrates that region 3 is nonempty, we believe that there are uncertain systems that are both RS$(\mathbf{\Delta}_C)$ and QS$(\mathbf{\Delta}_R)$ but not QS$(\mathbf{\Delta}_C)$.

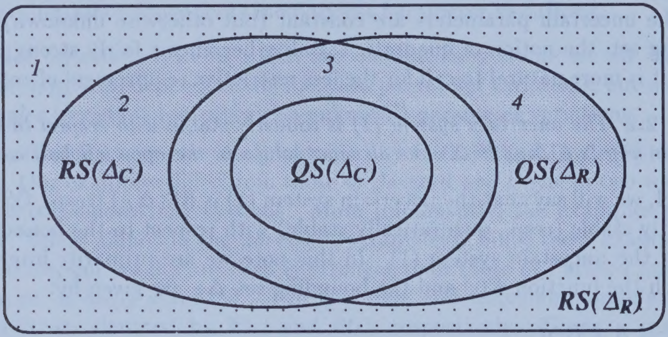


Figure 1: Relation among the various stability notions.

3 Examples

3.1 RS$(\mathbf{\Delta}_C)$ but not QS$(\mathbf{\Delta}_R)$

We begin this section by showing that there are norm bounded uncertain systems that are RS$(\mathbf{\Delta}_C)$ but not QS$(\mathbf{\Delta}_R)$, i.e. region 2 in Figure 1 is nonempty. Of course, in the view of Theorem 2.1, such systems must have at least two blocks of structured uncertainty.

Let Σ_1 denote the following uncertain system with structured norm bounded uncertainty:

$$\dot{x} = \overbrace{\begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -ab & -(a+b) \end{bmatrix}}^{A} x + \overbrace{\begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & kb \end{bmatrix}}^{B} \Delta \overbrace{\begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}}^{C} x, \tag{5}$$

where a, b, and $k > 0$ are given real numbers and

$$\Delta \in \mathbf{\Delta}_E := \{\text{block\_diag}(\Delta_1, \Delta_2) : \text{ for } i = 1, 2;\ \Delta_i \in \boldsymbol{E} \text{ and } |\Delta_i| \le 1\}. \tag{6}$$

Proposition 3.1 *Let a, b, and $k > 0$ be given real numbers and consider the uncertain system Σ_1 defined in (5–6). Then, Σ_1 is* QS($\mathbf{\Delta}_R$) *if and only if $a > 0$, $b > 0$, and $k < 1 + a/b$.*

Proof. *Necessity.* We first show that the conditions $a > 0$, $b > 0$, and $k < 1 + a/b$ are necessary for Σ_1 to be QS($\mathbf{\Delta}_R$). The necessity of the conditions $a > 0$ and $b > 0$ is immediate, for the eigenvalues of A are $\{-1, -a, -b\}$. Now, suppose that these two conditions are satisfied and that $k \geq 1 + a/b$. We will show that Σ_1 cannot be QS($\mathbf{\Delta}_R$). That is, given any real matrix $P = P' > 0$, there exists a real vector $\hat{x} \neq 0$ and a perturbation $\hat{\Delta} \in \mathbf{\Delta}_R$ such that the following inequality holds:

$$\phi(\hat{x}, \hat{\Delta}) := \hat{x}'P(A + B\hat{\Delta}C)\hat{x} \geq 0, \tag{7}$$

where A, B, and C are given by (5).

Consider any $P \in \mathbf{R}^{3\times 3}$ with $P = P' := [p_{ij}] > 0$. Suppose first that $p_{23} \leq 0$. Let $\hat{x} := [0 \;\; 1 \;\; 0]'$ and note that

$$\phi(\hat{x}, \Delta) := -abp_{23} \geq 0,$$

for any $\Delta \in \mathbf{\Delta}_R$. Thus, (7) holds.

Now, suppose that $p_{23} > 0$. Let $\hat{\Delta} :=$ block_diag(δ, δ), where the scalar δ is chosen according to

$$\delta := \begin{cases} \sqrt{(a+b)/bk} & \text{if } p_{12} \geq 0 \\ -\sqrt{(a+b)/bk} & \text{if } p_{12} < 0. \end{cases} \tag{8}$$

Note that $\hat{\Delta} \in \mathbf{\Delta}_R$, for $k \geq 1 + a/b$. Define also $\hat{x} := [1 + a/b \;\; 0 \;\; k\delta]'$, and note that $\hat{x} \neq 0$. Using (5–8), we obtain that

$$\begin{aligned} \phi(\hat{x}, \hat{\Delta}) &= k(1 + a/b)\delta p_{12} + (k\delta)^2 p_{23} \\ &> k(1 + a/b)\delta p_{12} \\ &\geq 0, \end{aligned}$$

and the proof of necessity is complete.

Sufficiency. Now suppose that $a > 0$, $b > 0$, and $0 < k < 1 + a/b$. We will construct a quadratic Lyapunov function for the uncertain system Σ_1. Let us begin by defining

$$D := \text{block\_diag}(\sqrt{(a+b)/bk},\;\; 1) \tag{9}$$

$$G(s) := DC(sI - A)^{-1}BD^{-1}, \tag{10}$$

where A, B, and C are given in (5).

A simple calculation shows that

$$\|G\|_\infty := \sup_{\omega \in R} \overline{\sigma}(G(j\omega)) = \sqrt{bk/(a+b)} < 1.$$

Hence, there exists $P \in \mathbf{R}^{3\times 3}$ with $P = P' > 0$ such that the following matrix inequality holds (see, for example, [8]):

$$Q(P) := A'P + PA + PBD^{-2}B'P + C'D^2C < 0. \tag{11}$$

We now show that the function V, given by $V(x) = \frac{1}{2}x'Px$ is a Lyapunov function for Σ_1. Indeed, let $x \in \boldsymbol{R}^n$ and $\Delta \in \boldsymbol{\Delta}_R$ be given. Then, the Lyapunov derivative is given by

$$\phi(x, \Delta) = x'P(A + B\Delta C)x.$$

Let D be given by (9) and note that $\Delta = D^{-1}\Delta D$, for $\Delta = \text{block\_diag}(\Delta_1, \Delta_2)$ with Δ_i scalars. Using this fact, along with $\overline{\sigma}(\Delta) \leq 1$ and the Cauchy Schwarz inequality it follows that the Lyapunov derivative satisfies

$$\begin{aligned}\phi(x, \Delta) &= x'PAx + x'PBD^{-1}\Delta DCx \\ &\leq x'PAx + \|D^{-1}B'Px\|\|DCx\|.\end{aligned}$$

From this inequality and (11), a standard "completion of the square argument" shows that

$$\phi(x, \Delta) \leq \frac{1}{2}x'Q(P)x.$$

The proof is completed by observing that $Q(P) < 0$. ■

We are now ready to construct the desired example. Let us define first

$$H_1(s) := \frac{1}{s+1}, \quad H_2(s) := \frac{kbs}{(s+a)(s+b)},$$

and note that the uncertain system Σ_1 introduced in (5) can be described by the block diagram of Figure 2. Since all transfer functions in this block diagram are scalar, a straightforward application of Theorem 2.1 allows us conclude that Σ_1 is RS($\boldsymbol{\Delta}_C$) if and only if $a > 0$, $b > 0$, and $\|H_1H_2\|_\infty < 1$.

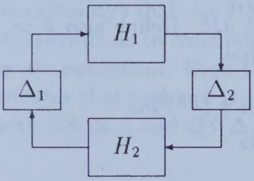


Figure 2: The uncertain system Σ_1.

Using this fact, along with Proposition 3.1 and the easily verifiable bound $\|H_1H_2\|_\infty < k/(1+a)$, we have the following example.

Example 3.1 *Let* $a > 0$, $b \geq 1$, *and* k *be given real numbers and consider the uncertain system* Σ_1 *defined in* (5–6). *Then, if* $1 + a/b \leq k \leq 1 + a$, Σ_1 *is* RS($\boldsymbol{\Delta}_C$) *but not* QS($\boldsymbol{\Delta}_R$)*; i.e.,* Σ_1 *lies in region 2 in the set diagram of Figure 1.*

It is interesting to note that when $a > 0$, $b \geq 1$ and $k \in [1 + a/b, 1 + a]$, there is a "time-varying" (real-valued) admissible perturbation $\Delta(\cdot)$ for which the uncertain system Σ_1 is not even *asymptotically stable*, yet the system is RS($\boldsymbol{\Delta}_C$). This is simply because time-varying uncertainties can "shift" the frequency responses of H_1 and H_2 so as to provide a large "equivalent loop gain". Of course, this phenomenon does not occur when the uncertainty is time-invariant. This was noticed in [6] for the discrete-time case.

To illustrate this point, let $\Delta(\cdot)$ denote a function of period T defined by

$$\Delta(t) := \begin{cases} \text{block\_diag}(0, \delta) & \text{if } 0 \le t < T/2 \\ \text{block\_diag}(-\delta, 0) & \text{if } T/2 \le t < T, \end{cases} \tag{12}$$

where the amplitude δ is free.

The instability of the system Σ_1 defined in (5), when the uncertainty is as in (12), can be inferred from its behavior at the instants of time $t = mT$ $(m = 0, 1, \ldots)$. Clearly, at those instants of time the state vector x evolves according to the difference equation

$$x(mT + T) = \Phi(a, b, k, T, \delta)x(mT), \tag{13}$$

where a detailed formula for the transition matrix Φ may be readily obtained from (5) and (12).

In Figure 3, we show the spectral radius of $\Phi(a, b, k, T, \delta)$ as a function of the amplitude δ. The other parameters are set to $a = 16$, $b = 1600$, $k = 16$, and $2\pi/T = 160$, so that the passing band of H_2 is very different from that of H_1. Note that if $\delta \ge 0.5$ (approximately) then Φ has at least one eigenvalue outside the unit disk. Hence, for these time-varying perturbations, the system Σ_1 is not asymptotically stable, which implies that it fails to be QS($\mathbf{\Delta}_R$). Of course, since $k = 16 > 1.01 = 1 + a/b$, this can also be seen from Proposition 3.1. On the other hand, since $k = 16 < 1 + a = 17$, it follows from Example 3.1 that Σ_1 is RS($\mathbf{\Delta}_C$)—actually robust stability is preserved for complex perturbations at least as large as 1.0625.

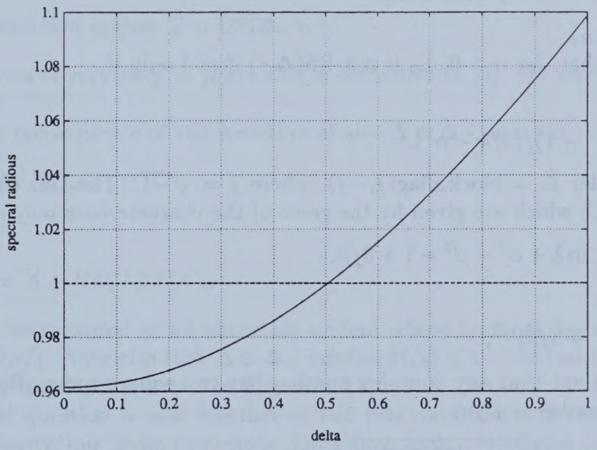


Figure 3: Spectral radius of Φ.

3.2 QS($\mathbf{\Delta}_R$) but not RS($\mathbf{\Delta}_C$)

We now show that there are norm bounded uncertain systems that are QS($\mathbf{\Delta}_R$) but not RS($\mathbf{\Delta}_C$), i.e. region 4 in Figure 1 is nonempty. Again, in view of Theorem 2.1, such systems must contain at least two blocks of uncertainty.

Let Σ_2 denote the uncertain system

$$\dot{x} = \overbrace{\begin{bmatrix} -\alpha & \beta \\ -\beta & -\alpha \end{bmatrix}}^{A} x + \overbrace{\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}}^{B} \Delta \overbrace{\begin{bmatrix} 0 & 1 \\ 1 & -1 \end{bmatrix}}^{C} x, \tag{14}$$

where α and β are real parameters, and

$$\Delta \in \boldsymbol{\Delta}_E := \{\text{block\_diag}(\Delta_1, \Delta_2) : \text{ for } i = 1, 2,\ \Delta_i \in \boldsymbol{E} \text{ and } |\Delta_i| \leq 1\}. \tag{15}$$

We first show that, given any $\alpha > \sqrt{5}/2$ and any real number β, Σ_2 is $\text{QS}(\boldsymbol{\Delta}_R)$. To this end, consider the candidate Lyapunov function $V(x) = \frac{1}{2}x'x$. Let $\Delta \in \boldsymbol{\Delta}_R$ be given, then the derivative of $V(x)$ along any solution of Σ_2 is given by

$$\begin{aligned} \phi(x, \Delta) &= x'(A + B\Delta C)x \\ &= -\alpha(x_1^2 + x_2^2) + \Delta_1 x_1 x_2 + \Delta_2(x_1^2 - x_2^2) \\ &\leq -\alpha(x_1^2 + x_2^2) + |x_1||x_2| + |x_1^2 - x_2^2|. \end{aligned}$$

From this last inequality, one may readily show that, if $\sqrt{5}/2 < \alpha$ then

$$\phi(x, \Delta) < 0 \text{ for all } x \neq 0, \Delta \in \boldsymbol{\Delta}_R.$$

Hence Σ_2 is $\text{QS}(\boldsymbol{\Delta}_R)$.

We now show that, for $\alpha > 0$, Σ_2 is not $\text{RS}(\boldsymbol{\Delta}_C)$ if and only if

$$\alpha < 3/2 \tag{16}$$

$$\beta^2 \geq \alpha^2(\alpha^2 - 1)/(9/4 - \alpha^2). \tag{17}$$

To see this, consider $\hat{\Delta} = \text{block\_diag}(j, -j)$, where $j = \sqrt{-1}$. The two eigenvalues of $F(\hat{\Delta}) := A + B\hat{\Delta}C$, which are given by the zeros of the characteristic polynomial

$$p(\lambda) := \lambda^2 + 2\alpha\lambda + \alpha^2 + \beta^2 - 1 + 3j\beta,$$

are

$$-\alpha + [1 - \beta^2 - 3j\beta]^{1/2},$$

where it is understood that any complex number has two square roots. Recalling that for any complex number $u + jv$,

$$\Re(\sqrt{u + jv}) = \pm\sqrt{\left[u + \sqrt{u^2 + v^2}\right]/2},$$

the real parts of the eigenvalues of $F(\hat{\Delta})$, are given by

$$-\alpha \pm \sqrt{\left[1 - \beta^2 + \sqrt{[(1 - \beta^2)^2 + 9\beta^2]}\right]/2}$$

For $\alpha > 0$, one can readily show that one of these two real numbers is nonnegative, hence Σ_2 is not $\text{RS}(\boldsymbol{\Delta}_C)$, if and only if

$$\beta^2(9 - 4\alpha^2) \geq 4\alpha^2(\alpha^2 - 1).$$

Finally, note that (for $\alpha > 0$) this inequality holds if and only if (16) and (17) hold.

We can now state the following result.

Example 3.2 *Consider the uncertain system* Σ_2 *described by* (14–15). *Suppose* $\sqrt{5}/2 < \alpha < 3/2$ *and* $\beta^2 \geq \alpha^2(\alpha^2 - 1)/(9/4 - \alpha^2)$. *Then* Σ_2 *is* $\text{QS}(\boldsymbol{\Delta}_R)$ *but not* $\text{RS}(\boldsymbol{\Delta}_C)$; *i.e.,* Σ_2 *lies in region 4 of Figure 1.*

4 Open issues

The examples provided in this paper have several implications. Perhaps, the most important one concerns the characterization of the notion of quadratic stability via an $\mathcal{H}_\infty$ condition with constant diagonal scaling. To elaborate on this point, we need the following result.

Lemma 4.1 *Let Σ denote an uncertain system with norm bounded structured uncertainty as defined in (1), (3), and (4). Let $G(s) := C(sI - A)^{-1}B$. Suppose that the matrix A has all eigenvalues in the open left half complex plane. Further, assume that there exist $r-1$ positive numbers $d_1, d_2, \dots, d_{r-1}$ such that*

$$\|D_o G D_i^{-1}\|_\infty < 1, \tag{18}$$

where

$$D_i := \text{block\_diag}(d_1 I_{m_1}, \dots, d_{r-1} I_{m_{r-1}}, I_{m_r}),$$
$$D_o := \text{block\_diag}(d_1 I_{p_1}, \dots, d_{r-1} I_{p_{r-1}}, I_{p_r}).$$

Then, the uncertain system Σ is $\mathrm{QS}(\mathbf{\Delta}_C)$.

This lemma is very easy to prove and is contained in [5]. For completeness, we give here a proof.

Proof. As a consequence of the structure of any $\Delta \in \mathbf{\Delta}_C$, one has

$$\Delta = D_i^{-1} \Delta D_o;$$

hence

$$F(\Delta) = A + B D_i^{-1} \Delta D_o C.$$

Thus Σ can be regarded as an uncertain system whose nominal description is given by $(A, BD_i^{-1}, D_o C)$. Note also that $\Delta \in \mathbf{\Delta}_C$ implies $\overline{\sigma}(\Delta) \le 1$. The result now follows from (18), and the implication $4 \Rightarrow 1$ in Theorem 2.1. ■

A natural question is now whether or not this condition is necessary for quadratic stability. Clearly, our second example shows that such a condition is not necessary for $\mathrm{QS}(\mathbf{\Delta}_R)$. If this were the case, quadratic stability for real perturbations would imply quadratic stability for complex perturbations. An open problem is to determine if the condition is necessary for $\mathrm{QS}(\mathbf{\Delta}_C)$. It is worth pointing out that, if there are at most two blocks of uncertainty, the answer to this question is in the affirmative. See, [6] for this result in a discrete time context, and [7] for its continuous-time counterpart.

Finally, it is important to note that these issues extend to *synthesis* problems with *linear* controllers. That is the problem of finding finite-dimensional and linear time invariant controllers that render a given uncertain system $\mathrm{QS}(\mathbf{\Delta}_E)$. This is simply because, without further assumptions on problem data, an analysis problem may be thought as a synthesis problem with zero control input matrix.

References

[1] P. Dorato (Editor), "Robust Control," IEEE Press: New York, 1987.

[2] P. Dorato and R. K. Yedavalli (Editors), "Recent Advances in Robust Control. A volume of selected conference papers," IEEE Press: New York, 1990.

[3] D. Hinrichsen and A. J. Pritchard, "Stability radii of linear systems," *Systems and Contr. Lett.*, vol. 7, pp. 1-10, 1986.

[4] P. P. Khargonekar, I. R. Petersen, and K. Zhou, "Robust stabilization of uncertain linear systems: quadratic stabilizability and $\mathcal{H}_\infty$ control theory," *IEEE Trans. on Automat. Contr.*, vol. 35, no. 3, pp. 356-361, 1990.

[5] A. Packard, Ph.D. dissertation, University of California, Berkeley, 1988.

[6] A. Packard and J. C. Doyle, "Quadratic stability with real and complex perturbations," *IEEE Trans. on Automat. Contr.*, vol. 35, no. 2, pp. 198-201, 1990.

[7] M. A. Rotea, "Systems with norm bounded uncertainties: quadratic stability with complex perturbations," in preparation.

[8] K. Zhou and P. P. Khargonekar, "An algebraic Riccati equation approach to $\mathcal{H}_\infty$ optimization," *Systems and Contr. Lett.*, vol. 11, pp. 85-91, 1988.

Robustness Analysis of Nonlinear Systems

Jeff S. Shamma
Department of Electrical Engineering
University of Minnesota
Minneapolis, MN 55455

Abstract

In this paper, we consider the robust stability of nonlinear feedback systems subject to nonlinear perturbations. For the special class of fading memory systems, we derive necessary and sufficient conditions for robust stability in the presence of a single block of unstructured dynamic uncertainty. We comment on possible extensions to block diagonal structured dynamic uncertainty.

1 Introduction

A common paradigm in robust stabilization is that of unstructured dynamic uncertainy. In this framework, the deviation from the nominal plant model is characterized by a perturbation representing an unknown but bounded dynamical system. One example is the so called "additive plant uncertainty" family given by $\mathcal{P}_{\text{add}} = \{P : P = P_o + \Delta W\}$. This family of plants may be given the following interpretation. The nominal plant is represented by P_o. The "unstructured uncertainty", Δ, is a dynamical system which is known only to be stable and satisfy a given norm bound. Finally the weighting, W, is a known stable dynamical system which "shapes and normalizes" the effects of Δ. The question of robust stabilization then takes the form "under what conditions does a compensator which stabilizes P_o also stabilize every P in $\mathcal{P}_{\text{add}}$?"

The theory for robust stability analysis for linear time-invariant systems subject to unstuctured uncertainty is well developed (e.g., [2,3,5,7,6,8,9,13]). For nonlinear systems, a standard tool for stability analysis is the small-gain theorem [4,17]. In this paper, we review recent results [14,15] which show that for the class of nonlinear systems with *fading-memory*, the small-gain theorem is in fact necessary—thus non-conservative—when considering nonlinear plant families characterized by a norm-bounded perturbation. Intuitively, the fading memory property means that the current output depends on the recent inputs and not the remote past. Thus, fading-memory is a reasonable assumption for many physical systems. Furthermore, we discuss possible extensions to block diagonal structured dynamic uncertainty.

The remainder of this paper is organized as follows. In Section 2, we review the results of [14,15] which state necessary and sufficient conditions for robust stability of nonlinear systems subject to a single block of unstructured uncertainty. In Section 3, we discuss possible extensions to diagonal structured uncertainty.

Notation. In this paper, we will consider both discrete-time and continuous-time systems with several notions of finite-gain stability (cf., [4,16]). Towards this end, the symbol $\mathcal{V}$ is used to denote any *one* of the following normed signal spaces: ℓ^p or $\mathcal{L}^p$ with $p \in [1, \infty)$, $\mathcal{L}_0^\infty$, and c_0, where

$$\mathcal{L}_0^\infty \stackrel{\text{def}}{=} \left\{ f \in \mathcal{L}^\infty : \lim_{t\to\infty} \sup_{\tau \geq t} |f(t)| = 0 \right\},$$

$$c_0 \stackrel{\text{def}}{=} \left\{ f \in \ell^\infty : \lim_{n\to\infty} |f(n)| = 0 \right\}.$$

The spaces $\mathcal{L}^p$ and ℓ^p are equipped with the usual norms, all denoted $\|\cdot\|$. The spaces $\mathcal{L}_0^\infty$ and c_0 are equipped with the usual "supremum" norms, also denoted $\|\cdot\|$. The symbol $\mathcal{T}$ is used to denote either $\mathcal{R}^+$ or $\mathcal{Z}^+$. Occasionally, it will be necessary to specify or restrict the particular definition of $\mathcal{V}$ and $\mathcal{T}$.

For $T \in \mathcal{T}$, S_T denotes the T-shift (time-delay) operator, and P_T denotes the truncation operator. The extended space, $\mathcal{V}_e$, is defined as $\mathcal{V}_e \overset{\text{def}}{=} \{f : \mathcal{V} \to \mathcal{R}^n : P_T f \in \mathcal{V}, \forall T \in \mathcal{T}\}$. The set of all $f \in \mathcal{V}_e$ with $f \notin \mathcal{V}$ is denoted by $\mathcal{V}_e \backslash \mathcal{V}$.

The definitions of causal, time-invariant, locally Lipshitz continuous, stable, incrementally stable, and causally invertible are as in [16]. The definitions of stability over $\mathcal{L}_0^\infty$ and c_0 are somewhat non-standard. For causal operators, stability over $\mathcal{L}_0^\infty$ implies stability over $\mathcal{L}^\infty$ with the same induced norm. The main difference is that stability over $\mathcal{L}_0^\infty$ implies a notion of asymptotic stability in a bounded-input/bounded-output setting. Similar arguments hold for stability over c_0.

Henceforth, all operators are assumed to be causal. The set of all *stable* NLTV operators $H: \mathcal{V}_e \longrightarrow \mathcal{V}_e$ is denoted $\mathcal{S}_{NL}$. The subset of operators in $\mathcal{S}_{NL}$ which are *linear* (and possibly time-varying) is denoted $\mathcal{S}_L$.

2 Unstructured Dynamic Uncertainty

2.1 Fading Memory

A notion of fading-memory is the central focus of this paper. Intuitively, fading-memory implies that the effects of a finite-duration of input eventually vanish. A definition of fading-memory tailored to time-invariant systems over ℓ^∞ or $\mathcal{L}^\infty$ signals was given in [1]. An alternate definition for time-varying systems over ℓ^2 was given in [14]. The following definitions generalize those in [14] to time-varying systems over $\mathcal{V}$.

Definition 2.1 *An operator $H \in \mathcal{S}_{NL}$ is said to have* **pointwise finite-memory** *if there exists a function $FM(\cdot,\cdot;H) : \mathcal{V} \times \mathcal{T} \to \mathcal{T}$ such that for all $f \in \mathcal{V}$ and $t \in \mathcal{T}$*

1. $FM(f,t;H) \geq t$,
2. $FM(f,t;H) = FM(P_t f,t;H)$,
3. $(I - P_{FM(f,t;H)})Hf = (I - P_{FM(f,t;H)})H(I - P_t)f$.

Note that $FM(f,t;H)$ is causally dependent on f.

This definition of pointwise finite-memory is somewhat weaker than that in [14]. The definition in [14] requires that inputs over a given finite-duration are forgotten *uniformly* as follows:

Definition 2.2 *An operator $H \in \mathcal{S}_{NL}$ is said to have* **uniform finite-memory** *if there exists a function $FM(\cdot;H) : \mathcal{T} \to \mathcal{T}$ such that for all $f \in \mathcal{V}$ and $t \in \mathcal{T}$*

1. $FM(t;H) \geq t$,
2. $(I - P_{FM(t;H)})Hf = (I - P_{FM(t;H)})H(I - P_t)f$.

Fading-memory operators are now defined as follows:

Definition 2.3 *An operator $H \in \mathcal{S}_{NL}$ is said to have* **uniform (resp., pointwise) fading-memory** *if it can be approximated arbitrarily closely in norm by uniform (resp., pointwise) finite-memory operators.*

It turns out that the distinction between uniform and pointwise fading memory is important. For example, while all discrete-time linear systems exhibit uniform fading memory, this is not the case in continuous-time. However, continuous-time linear systems do exhibit pointwise fading memory. Furthermore, finite-dimensional continuous-time linear systems exhibit uniform fading memory.

Some useful consequences of these definitions are now presented. The proofs may be found in [14,15].

The following proposition states that pointwise finite-memory nonlinear operators are right-distributive over signals with sufficient time-separation.

Proposition 2.1 *Let H have pointwise finite-memory. Let $f_1 \in \mathcal{V}$ have supp(f_1) $\subset [0, T_1]$. Let $T_2 = FM(f_1, T_1; H)$. Then for all $f_2 \in \mathcal{V}$ with supp(f_2) $\subset (T_2, \infty)$,*

$$H(f_1 + f_2) = Hf_1 + Hf_2.$$

The following theorem characterizes how the effects of a finite-duration input eventually decay in uniform fading-memory operators.

Theorem 2.1 *For $H \in \mathcal{S}_{NL}$, the following statements are equivalent:*

- *(a) The operator H has uniform fading-memory.*
- *(b) Given any $\varepsilon > 0$, there exists a function $FM_\varepsilon(\cdot; H) : \mathcal{T} \to \mathcal{T}$ with $FM_\varepsilon(t; H) \geq t$ such that*

$$\left\| (I - P_{FM_\varepsilon(t;H)}) Hf - (I - P_{FM_\varepsilon(t;H)}) H (I - P_t) f \right\| \leq \varepsilon \|f\|,$$

  for all $f \in \mathcal{V}$ and $t \in \mathcal{T}$.

The following theorem states that all discrete-time linear operators have uniform fading-memory.

Theorem 2.2 *Let $\mathcal{V}$ denote ℓ^p with $p \in [1, \infty)$ or c_0. Then every $H \in \mathcal{S}_L$ has uniform fading-memory.*

This is not the case in continuous-time, as seen in the following example.

Example 2.1 A pointwise but non-uniform fading-memory linear operator. Let $\mathcal{M}$ denote the set of $f \in \mathcal{L}^2[0,1)$ such that $t \mapsto f\left(\frac{t-1}{t}\right) \in \mathcal{L}^2[1, \infty)$. Define $H : \mathcal{L}^2 \to \mathcal{L}^2$ by

$$(Hf)(t) = \begin{cases} 0, & 0 \leq t < 1; \\ 0, & t \geq 1, f|_{[0,1)} \notin \mathcal{M}; \\ f\left(\frac{t-1}{t}\right), & t \geq 1, f|_{[0,1)} \in \mathcal{M}. \end{cases}$$

In words, the operator H takes the input over the time-interval $[0, 1)$ and streches it over the time-interval $[1, \infty)$.

From Theorem 2.1, it is clear that H does not have uniform fading-memory. However, H does have *pointwise* fading-memory. To see this, we will construct a pointwise finite-memory approximant, $\tilde{H}$, of H as follows. Given any $\varepsilon > 0$, define $\tilde{H}$ by

$$(\tilde{H}f)(t) = \begin{cases} 0, & 0 \leq t < 1; \\ 0, & t \geq 1, f|_{[0,1)} \notin \mathcal{M}; \\ f\left(\frac{t-1}{t}\right), & t \geq 1, f|_{[0,1)} \in \mathcal{M}, \left\|P_{\frac{t-1}{t}} f\right\| \leq (1-\varepsilon) \left\|f|_{[0,1)}\right\|; \\ 0, & \left\|P_{\frac{t-1}{t}} f\right\| > \varepsilon \left\|f|_{[0,1)}\right\|. \end{cases}$$

From this definition, it is easy to see that $\left\|H - \tilde{H}\right\| \leq \varepsilon$. The main idea is that for any signal f, $\tilde{H}$ truncates the portion in the interval $[0, 1)$ which will be streched to $[1, \infty)$. However, the portion of $[0, 1)$ to be truncated is ***signal-dependent***, hence the restriction to pointwise fading-memory only. ∎

The above construction of a pointwise fading-memory approximant to a linear system provides the main insight for the following theorem.

Theorem 2.3 *For $H \in \mathcal{S}_{NL}$, the following statements are equivalent:*

(a) The operator H has pointwise fading-memory.

(b) Given any $\varepsilon > 0$ the following condition holds. There exists a function $FM_\varepsilon(\cdot, \cdot; H) : \mathcal{V} \times \mathcal{T} \to \mathcal{T}$ such that for all $f \in \mathcal{V}$ and $t \in \mathcal{T}$

1. $FM_\varepsilon(f, t; H) \geq t$,
2. $FM_\varepsilon(f, t; H) = FM_\varepsilon(P_t f, t; H)$,
3. $\left\|(I - P_{FM_\varepsilon(f,t;H)})Hf - (I - P_{FM_\varepsilon(f,t;H)})H(I - P_t)f\right\| \leq \varepsilon \|f\|$.

Corollary 2.1 *Let $\mathcal{V}$ denote $\mathcal{L}^p$ with $p \in [1, \infty)$ or $\mathcal{L}_0^\infty$. Then every S_L has pointwise fading-memory.*

Finite-dimensional linear systems, however, always exhibit uniform fading memory.

Theorem 2.4 *The input-to-state mapping of a continuous-time linear system has uniform fading memory.*

2.2 Necessary Conditions for Robust Stability

The main results of [14] deal with uniform fading-memory systems over ℓ^2. In this section, these results are extended to encompass pointwise fading-memory continuous-time and discrete-time systems over any signal-space in $\mathcal{V}$.

The main results are the following. See [15] for proofs.

Theorem 2.5 *Let $\mathcal{V}$ denote any one of the $\mathcal{L}_p$ or ℓ_p spaces with $p \in [1, \infty)$. For $H \in \mathcal{S}_L$, let $Q \in \mathcal{S}_L$ be called "admissible" if the operator $I - QH$ is causally invertible and $\|Q\| < 1$. Then the operator $I - QH$ has a* **stable** *inverse for all admissible $Q \in \mathcal{S}_L$ if and only if*

$$\inf_{t \in \mathcal{T}} \|HS_t\| \leq 1.$$

Definition 2.4 *Let $H \in \mathcal{S}_{NL}$ satisfy*

$$\inf_{t \in \mathcal{T}} \|HS_t\| = \gamma.$$

Then H is said to have the **uniformity in norm-excitation** *(UINE) property if the following condition holds. Given any $\varepsilon \in (0, \gamma)$, there exists a sequence $\{f_k\} \subset \mathcal{V}$ such that*

1. $\|HS_k f_k\| \geq (\gamma - \varepsilon) \|f_k\|, \quad \forall k \in \mathcal{Z}^+$.
2. $\sup_k \|f_k\| < \infty$.
3. $\inf_k \|f_k\| > 0$.

The UINE property states that to approach the operator gain, $||HS_k||$, does not require injecting signals become arbitrarily large or small as $k \longrightarrow \infty$. Note that by the homogeneity property of linear systems, any $H \in \mathcal{S}_L$ has the UINE property. Furthermore, any nonlinear $H \in \mathcal{S}_{NL}$ which is *time-invariant* also has the UINE property.

Theorem 2.6 *Let $\mathcal{V}$ denote any one of the $\mathcal{L}_p$ or ℓ_p spaces with $p \in [1, \infty)$. Let $H \in \mathcal{S}_{NL}$ be locally Lipshitz continuous and have pointwise fading-memory and the UINE property. Let $Q \in \mathcal{S}_{NL}$ be called "admissible" if the operator $(I - QH)$ is causally invertible, $(I - QH)^{-1}$ is locally Lipshitz continuous, Q has pointwise fading-memory, and $||Q|| < 1$. Then the operator $I - QH$ has a* **stable** *inverse for all admissible $Q \in \mathcal{S}_{NL}$* **and**

$$\sup_{Q \text{ admissible}} \left\|(I - QH)^{-1}\right\| < \infty$$

if and only if

$$\inf_{t \in T} ||HS_t|| < 1.$$

A key difference between Theorem 2.5 and Theorem 2.6 is that Theorem 2.6 provides conditions for uniform stability. The reason for this is that in case H is nonlinear (as in Theorem 2.6), a *destabilizing* Q is constructed for H with finite memory but *not* fading memory. Rather, when H has fading memory, it is the uniformity in stability which is violated.

The local Lipshitz continuity assumption on H is not necessary to prove the necessity part of Theorem 2.6. Rather, it is merely a sufficient condition to guarantee $(I - QH)^{-1}$ is locally Lipshitz continuous for any linear Q. This is needed in the sufficiency portion of the proof.

The main idea in the proof for finite-memory it to exploit the right-distributivity property of Proposition 2.1. In fact, one could use this property alone to define an even weaker notion of pointwise fading memory for which Theorem 2.6 still holds.

It turns out that for $\mathcal{V} = \mathcal{L}_0^\infty$ or c_0, it is possible to construct a *destabilizing* Q for fading memory H.

Theorem 2.7 *Let $\mathcal{V}$ denote either $\mathcal{L}_0^\infty$ or c_0. Let $H \in \mathcal{S}_{NL}$ have pointwise fading-memory and the UINE property. Let $Q \in \mathcal{S}_{NL}$ be called "admissible" if the operator $I - QH$ is causally invertible and $||Q|| < 1$. Then the operator $I - QH$ has a* **stable** *inverse for all admissible $Q \in \mathcal{S}_L$ only if*

$$\inf_{t \in T} ||HS_t|| \leq 1.$$

In fact, it is not shown that $(I - QH)^{-1}$ is an unbounded operator. Rather, it is the asymptotic stability which is violated.

At a glance, it would seem that the stated necessary conditions is also sufficient for robust invertibility. However, standard small gain arguments would only assure that the operator $(I - QH)^{-1}$ is stable over $\mathcal{L}^\infty$ and not necessarily $\mathcal{L}_0^\infty$. Some additional work is required to guarantee asymptotic stability.

Theorem 2.8 *Let $\mathcal{V}$ denote either $\mathcal{L}_0^\infty$ or c_0. Let $G \in \mathcal{S}_{NL}$ have pointwise fading memory and $||G|| < 1$. Let $(I - G)$ be causally invertible with $(I - G)^{-1}$ continuous, i.e.,*

$$f_n \to f_o \quad \Rightarrow (I - G)^{-1} f_n \to (I - G)^{-1} f_o.$$

Then $(I - G)^{-1}$ is stable over $\mathcal{V}$.

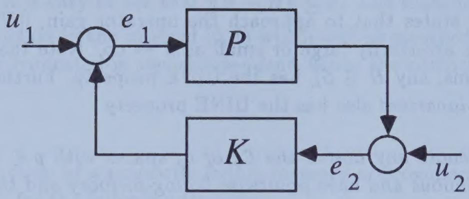


Figure 1: Block Diagram for Robust Stabilization

We now consider the block diagram of Figure 1. Some preliminary assumptions and definitions are as follows. The plant, P, and compensator, K, are operators on $\mathcal{V}_e$. This feedback system is said to be *well-posed* (cf., [16]) if given any $(u_1, u_2) \in \mathcal{V}_e \times \mathcal{V}_e$, there exist unique $(e_1, e_2) \in \mathcal{V}_e \times \mathcal{V}_e$ which satisfy

$$\begin{aligned} e_1 &= u_1 + Ke_2 \\ e_2 &= u_2 + Pe_1. \end{aligned}$$

such that the mapping $\Phi(P, K) : \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \mapsto \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$ is causal. Assuming well-posedness, the compensator, K, is said to *(incrementally) stabilize* the plant, P, if $\Phi(P, K)$ is (incrementally) stable. The compensator K is said to *pointwise fading memory (incrementally) stabilize* P if $\Phi(P, K)$ is (incrementally) stable and has pointwise fading memory.

The previous stability theorems can be used to give necessary and sufficient conditions for robust stability as follows. Define the following *family* of plants:

$$\mathcal{P}_{\text{add}} \stackrel{\text{def}}{=} \{P : P = P_o + \Delta W\}$$

where $\Delta \in \mathcal{S}_{NL}$ with $||\Delta|| < 1$, and $W, W^{-1} \in \mathcal{S}_{NL}$ with W incrementally stable. We assume that P_{add} is such that any causal compensator results in a well-posed feedback system for every $P \in \mathcal{P}_{\text{add}}$.

The problem of robust stabilization is under what conditions does a compensator, K, which stabilizes P_o also stabilize every $P \in \mathcal{P}_{\text{add}}$ with

$$\sup_{P \in \mathcal{P}_{\text{add}}} ||\Phi(P, K)|| < \infty.$$

In this case, the compensator, K, is said to *uniformly robustly stabilize* the family $\mathcal{P}_{\text{add}}$.

Theorem 2.9 *Let $\mathcal{V}$ denote any one of the $\mathcal{L}_p$ or ℓ_p spaces with $p \in [1, \infty)$. Consider the plant family $\mathcal{P}_{\text{add}}$. Let the compensator, K, pointwise fading memory incrementally stabilize P_o such that $WK(I - P_oK)^{-1}$ satisfies the UINE property. Then K uniformly robustly stabilizes the family $\mathcal{P}_{\text{add}}$ if and only if*

$$\inf_{t \in T} \left\| WK(I - P_oK)^{-1} S_t \right\| < 1.$$

In case $\mathcal{V}$ denotes $\mathcal{L}_0^\infty$ or c_0, Then K robustly stabilizes the family $\mathcal{P}_{\text{add}}$ only if

$$\inf_{t \in T} \left\| WK(I - P_oK)^{-1} S_t \right\| \leq 1.$$

It is noted that the requirement of incremental stability is for the sufficiency of the stated condition for robust stability. The necessity follows without recourse to incremental stability.

A key assumption in Theorem 2.9 is that the closed-loop operator $WK(I - P_oK)^{-1}$ exhibits pointwise fading memory. In case P is linear (possible time-varying), it is possible to parameterize all fading memory incrementally stabilizing compensators [15].

Definition 2.5 *Let $H : \mathcal{V}_e \to \mathcal{V}_e$. Then $H = ND^{-1}$ is said to be a* **right-coprime factor representation** *(r.c.f.r.) of H if*

1. *$N, D \in \mathcal{S}_{NL} \in \mathcal{S}_{NL}$.*
2. *D is causally invertible.*
3. *There exists an $F \in \mathcal{S}_{NL}$ such that*

$$F\begin{pmatrix} N \\ D \end{pmatrix} = I,$$

i.e., the matrix operator $\begin{pmatrix} N \\ D \end{pmatrix}$ has an stable left inverse.

Theorem 2.10 *Let $K_L = XY^{-1}$ be a linear compensator which stabilizes $P = NM^{-1}$, where XY^{-1} and NM^{-1} are r.c.f.r.'s. Then all pointwise fading memory incrementally stabilizing nonlinear compensators are parameterized by*

$$K = (X + MQ)(Y + NQ)^{-1}$$

with Q incrementally stable with pointwise fading memory.

A direct consequence of these results is that nonlinear controllers over no advantage over linear controllers for robust stabilization over $\mathcal{L}^2$ and ℓ^2. This has been shown in [11,12] when the nominal plant is linear time-invariant. The following holds for linear time-varying plants.

Theorem 2.11 *Let $\mathcal{V}$ denote either $\mathcal{L}^2$ or ℓ^2. Consider the plant family $\mathcal{P}_{add}$ with the added assumption that P_o and W are linear. Let P_o be stabilizable by a linear controller. If there exists nonlinear time-varying controller which (1) pointwise fading memory incrementally stabilizes P_o and (2) robustly stabilizes the family P_{add}, then there exists a linear time-varying controller which achieves the same goals (1) and (2).*

3 Structured Uncertainty

In this section, we comment on possible extensions of the results of Section 2 to systems with *structured uncertainty* [5,9,10]. The problem under consideration is as follows.

Question. *Let the N-input/N-output $H \in \mathcal{S}_{NL}$ have pointwise fading-memory and the UINE property. Let $Q \in \mathcal{S}_{NL}$ be called "admissible" if the operator $I - QH$ is causally invertible, $\|Q\| < 1$, and Q is* **diagonal**. *Under what conditions does $I - QH$ has a* **stable** *inverse for all admissible $Q \in \mathcal{S}_L$?*

Reference [5] answers this question in the case of $\mathcal{L}^2$ stability with H linear time-invariant. The case of $\mathcal{L}^\infty$ stability is solved in [9] for H linear time-invariant, and in [10] for H linear time-varying.

A main idea in [9,10] is to show (modulo a non-degeneracy assumption) that there exists a constant diagonal scaling such that each row of H has a norm greater than one. From this condition, an admissible destabilizing Q is constructed. In this section we show that in case H is nonlinear, there exists such a diagonal scaling. However, we provide a counter-example which shows that this is not sufficient to construct a destabilizing Q.

In the remainder of this section, $\mathcal{V}$ denotes either $\mathcal{L}_0^\infty$ or c_0.

First, the following notation is established. Let $\mathbf{R}_i$ denote the "ith row operator. That is,

$$\mathbf{R}_i \begin{pmatrix} f_1 \\ \vdots \\ f_N \end{pmatrix} \stackrel{\text{def}}{=} \begin{pmatrix} 0 \\ \vdots \\ f_i \\ \vdots \\ 0 \end{pmatrix}.$$

We make the following assumptions on H:
Assumption 3.1 *The N-input/N-output $H \in \mathcal{S}_{NL}$ is such that*

$$\mathbf{R}_i H \mathbf{R}_j \neq 0, \quad \forall i, j.$$

Furthermore, for any sequence $\{D_n\}$ of $N \times N$ diagonal matrices such that $D_n \to D_*$, one has that

$$\|HD_n - HD_*\| \to 0.$$

Proposition 3.1 *Let H be such that $\|D^{-1}HD\| \geq 1$ for all diagonal D with positive elements. Then there exists a D_* such that*

$$\left\|\mathbf{R}_i D_*^{-1} H D_*\right\| \geq 1, \quad \forall i.$$

Proof Define the mapping T over positive diagonal matrices by

$$[T(D)]_i = \frac{\|\mathbf{R}_i HD\|}{\|HD\|},$$

where $[\cdot]_i$ denotes the ith diagonal element. Now define the sequence $\{D_n\}$ of positive diagonal matrices by the recursion

$$D_{n+1} = T(D_n), \quad D_0 = 0.$$

Note that each D_n has at least one diagonal element equal to one. By Assumption 3.1, $\|HD_n\|$ is bounded away from zero. Furthermore, the matrices D_n all have diagonal elements which take values in the interval $(0, 1]$. SInce it is a bounded sequence, let D_* be the limit of a convergent subsequence. By the assumed continuity property of H in Assumption 3.1, it follows that

$$[D_*]_i \|HD\| = \|\mathbf{R}_i HD_*\|, \quad \forall i.$$

Thus,

$$\|HD\| = \left\|\mathbf{R}_i \frac{1}{[D_*]_i} HD_*\right\|, \quad \forall i.$$

However by hypothesis,

$$\left\|D_*^{-1}HD_*\right\| = \max_i \left\|\mathbf{R}_i \frac{1}{[D_*]_i} HD_*\right\| \geq 1,$$

which completes the proof. ∎

The assumed condition that $\|D^{-1}HD\| \geq 1$ is necessary for the existence of a *linear* Q to destabilize $(I - QH)^{-1}$. The restriction to linear Q's is minor in that only linear Q's were used to lead to the necessary conditions for robust stability in Section 2. That is, there was no need to resort to nonlinear Q's to establish small-gain necessity.

We now provide a counter-example which shows that the row condition

$$\left\| \mathbf{R}_i D_*^{-1} H D_* \right\| > 1$$

is *not* sufficient to construct a destabilizing Q. This is in contrast to the linear case [9,10].

Let the 2×2 nonlinear $H \in \mathcal{S}_{NL}$ acting on c_0 be defined by:

$$(Hf)(n) \overset{\text{def}}{=} \begin{pmatrix} 61\mathcal{D}(f_2(n)/60) \\ \mathcal{N}(51 f_1(n))/50 \end{pmatrix},$$

where $\mathcal{N}$ and $\mathcal{D}$ denote the standard unity saturation and deadzone functions. Note that $\|\mathbf{R}_1 H\| = 61/60$ and $\|\mathbf{R}_2 H\| = 51/50$. We will show that $(I - QH)^{-1}$ is stable over c_0 for any admissible diagonal Q with $\|Q\| < 1$. It will be convenient to write the equation $(I - QH)f = g$ to

$$f_1 - Q_1(61\mathcal{D}(f_2/60)) = g_1,$$

$$f_2 - Q_2(\mathcal{N}(51 f_1)/50) = g_2.$$

We first show that $g \in c_0$ implies $f \in c_0$. Let $g \in c_0$. Since $\lim_n g_2(n) = 0$, it follows that $\lim_n f_2(n) \leq (1/50)$. Using this fact in conjuction with $\lim_n g_1(n) = 0$ leads to $\lim_n f_1(n) = 0$. This in turn implies $\lim_n f_2(n) = 0$.

Thus in order for $(I - QH)^{-1}$ to be unstable over c_0, it must be unbounded. That is, there exist sequences $\{f^n\}, \{g^n\} \subset c_0$ such that

$$f^n = (I - QH)^{-1} g^n$$

and $\|f^n\| / \|g^n\| \to \infty$. Using elementary arguments, it can be shown that one can make $\|f^n\| / \|g^n\| \to \infty$ only by letting $\|g^n\| \to 0$. In this case, $\lim_n \|f_2^n\| \leq 1/50$. Then for sufficiently large n, $f_1^n = g_1^n$, which in turn implies $\|f_2^n\| \leq \|g_2^n\| + \|g_1^n\| / 50$. Thus, the ratio $\|f^n\| / \|g^n\|$ is well behaved.

The key in this example is that although each row of H has a norm greater than one, the size of the "exciting signal" in each case is very different. In order to excite the first row requires f_2 very large. In order to excite the second row requires f_1 very small. This dichotomy in the sizes along with the skew construction of H leads to the desired counterexample.

Acknowledgments This research was partially supported by an AFOSR Summer Faculty Fellowship at the Armament Laboratory at Eglin Air Force Base, FL 32542.

References

[1] S. Boyd and L.O. Chua, "Fading Memory and the Problem of Approximating Nonlinear Operators with Volterra Series," *IEEE Transactions on Circuits and Systems*, Vol. CAS–32, 1985, pp. 1150–1161.

[2] M.J. Chen and C.A. Desoer, "Necessary and Sufficient Condition for Robust Stability of Linear Distributed Feedback Systems," *International Journal of Control*, Vol. 35, No. 2, 1982, pp. 255–267.

[3] M.A. Dahleh and Y. Ohta, "A Necessary and Sufficient Condition for Robust BIBO Stability," *Systems & Control Letters* **11**, 1988, pp. 271–275.

[4] C.A. Desoer and M. Vidyasagar, *Feedback Systems: Input-Output Properties*, Academic Press, New York, 1975.

[5] J. Doyle, "Analysis of Feedback Systems with Structured Uncertainties," *IEE Proceedings*, Vol. 129, Part D, No. 6, 1982, pp. 242–250.

[6] J.C. Doyle, J.E. Wall, and G. Stein, "Performance and Robustness Analysis for Structured Uncertainty," *Proceedings of the 21st IEEE Conference on Decision and Control*, 1982, pp. 629–636.

[7] J.C. Doyle and G. Stein, "Multivariable Feedback Design: Concepts for a Classical/Modern Synthesis," *IEEE Transactions on Automatic Control*, Vol. AC–26, 1981, pp. 4–16.

[8] T.T. Georgiou and M. Smith, "Optimal Robustness in the Gap Metric," *IEEE Transactions on Automatic Control*, Vol. AC–35, 1990, pp. 673–686.

[9] M. Khammash and J.B. Pearson, "Performance Robustness of Discrete-Time Systems with Structured Uncertainty," *IEEE Transactions on Automatic Control*, Vol. AC–36, No. 4, 1991, pp. 398–412.

[10] M. Khammash, "Necessary and and Sufficient Conditions for the Robustness of Time-Varying Systems with Applications to Sampled-Data Systems," submitted to *IEEE Transactions on Automatic Control*, 1991.

[11] P.P. Khargonekar, T.T. Georgiou, and A.M. Pascoal, "On the Robust Stabilization of Linear Time-Invariant Plants with Unstructured Uncertainty," *IEEE Transactions on Automatic Control*, Vol. AC–32, 1987, pp. 201–207.

[12] K. Poolla and T. Ting, "Nonlinear Time-Varying Controllers for Robust Stabilization," *IEEE Transactions on Automatic Control*, Vol. AC–32, 1987, pp. 195-200.

[13] J.S. Shamma and M.A. Dahleh, "Time-Varying vs. Time-Invariant Compensation for Rejection of Persistent Bounded Disturbances and Robust Stabilization," *IEEE Transactions on Automatic Control*, July 1991.

[14] J.S. Shamma, "The Necessity of the Small Gain Theorem for Time-Varying and Nonlinear Systems," *IEEE Transactions on Automatic Control*, October 1991.

[15] J.S. Shamma and R. Zhao, "Fading Memory Feedback Systems and Robust Stability," submitted to *Automatica*, 1991.

[16] J.C. Willems, *The Analysis of Feedback Systems*, MIT Press, Cambridge, MA, 1971.

[17] G. Zames, "On the Input-Output Stability of Time-Varying Nonlinear Feedback Systems. Part I: Conditions Using Concepts of Loop Gain, Conicity, and Positivity," *IEEE Transactions on Automatic Control*, Vol. AC-11, No. 2, 1966, pp. 228–238.

Finite Horizon $\mathcal{H}_\infty$ Digital Control for Sampled-Data Systems *

Krishan M. Nagpal‡, Weiqian Sun†, and Pramod P. Khargonekar†

Abstract

In this paper, finite horizon $\mathcal{H}_\infty$ control problems for digital control of sampled-data systems are studied. Necessary and sufficient conditions are obtained for the existence of controllers that satisfy a specified $\mathcal{H}_\infty$ performance bound. These conditions are stated using two Riccati differential equations with jumps. When these conditions hold, explicit formulae for the controller satisfying the $\mathcal{H}_\infty$ performance bound are also given.

1 Introduction

Suppose we are given continuous-time linear time-invariant plant and we are interested in designing a digital controller. A digital controller is a discrete-time system which is connected to the plant by means of A/D and D/A (zero order hold) devices. Thus, the control input is piecewise constant. In this paper, we investigate the problem of designing digital controllers that satisfy given *finite horizon* $\mathcal{H}_\infty$ performance requirement . The $\mathcal{H}_\infty$ performance requirements are defined directly in terms of the continuous-time exogenous inputs and regulated outputs, and thus take inter-sample behavior into account.

During the last few years, analysis and design of sampled-data control systems has been a subject of intense research activity. A number of papers have appeared recently that deal with robust and optimal control of sampled-data systems, e.g., [1, 2, 3, 4, 5, 8, 10, 16, 20, 21, 22, 23]. (See the recent survey paper by Francis [7] which also contains a large bibliography.) This work is closely related to the work on $\mathcal{H}_\infty$ optimal control of sampled-data systems by Bamieh and Pearson [1], Başar [2], Başar and Bernhard [3], Bernhard [4], Chen and Francis [5], Hara and Kabamba [8], Tadmor [20] Toivonen [21, 22]. In [1, 8, 21, 22], this problem is solved be transforming the problem into an equivalent *discrete-time $\mathcal{H}_\infty$ controller design problem* for a suitably constructed discrete-time system. Game theoretic approach is taken in [2, 3, 4] for similar problems, but the input is not constrained to be piecewise constant. Our results are most closely related to the work of Toivonen [22] where he has obtained results on the state-feedback case for the finite horizon digital control case.

---

[*] Supported in part by National Science Foundation under grants no. ECS-9001371, Airforce Office of Scientific Research under contract no. AFOSR- 90-0053, Army Research Office under grant no. DAAL03-90-G-0008.

[‡] Dept. of Mechanical Engineering, University of California, Berkeley, CA 94720.

[†] Dept. of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, MI 48109-2122.

Here we pursue our approach to these problems [18] and give solutions for both the full-information and output feedback cases. We give necessary and sufficient condition for the existence of a causal controller satisfying a specified $\mathcal{H}_\infty$ performance requirement. These necessary and sufficient conditions are in terms of existence of solutions to differential matrix Riccati equations. If these solutions exist, then we give formulae for filters/controllers that satisfy the given $\mathcal{H}_\infty$ performance requirement. The matrix Riccati equations are given explicitly in terms of the plant system matrices and no transformations are needed.

We end this introduction with some remarks on the notation most of which is standard: $\mathcal{R}$ denotes the set of real numbers, $\mathcal{R}^n$ denotes the n dimensional Euclidean space (identified with $n \times 1$ vectors of real numbers,) and $\mathcal{R}^{n \times m}$ denotes the set of all $n \times m$ real matrices. We will use A'(or v') to denote the transpose of the matrix A(or vector v,) $\rho(A) := \max_i |\lambda_i(A)|$ the spectral radius. The following norms will be used: $\|\eta\| := (\sum_1^n |\eta_i|^2)^{1/2}$ for $\eta \in \mathcal{R}^n$; $\|f\|_{[a,b]} := (\int_a^b \|f(t)\|^2 dt)^{1/2}$ if f is function of $t \in [a,b]$; $\|f\|_{(a,b)} := (\sum_{ih \in (a,b)} \|f(ih)\|^2)^{1/2}$ if f is function of discrete time ih, i an integer and $h > 0$. The corresponding square integrable (or summable) spaces will be denoted by $\mathcal{L}_2^n[a,b]$(or $\ell_2^n(a,b)$.) For simplicity, we will assume a uniform sampling rate, i.e. h is a fixed sampling period. For a matrix or vector function $f(t)$ which is right continuous and may be left discontinuous, we use $f(ih^-)$ to denote the value of $f(t)$ just before ih, i.e., $f(ih^-) := \lim_{\epsilon > 0, \epsilon \to 0} f(ih - \epsilon)$ assuming that the limit exists. The truncation operator π_τ is defined as $(\pi_\tau)f(t) = 0$ for $t < \tau$, $= f(t)$ for $t \geq \tau$. Finally $\Phi_A(s,t)$ is the state transition matrix associated with differential equation $\dot{x}(t) = Ax$.

2 Main Results

In this section, we consider the linear system G:

$$\dot{x}(t) = Ax(t) + Bw(t) + Eu(t), \ t \in [0,T] \tag{2.1}$$

$$x(0) = 0 \tag{2.2}$$

where $w(t) \in \mathcal{R}^m$ is the exogenous (or disturbance) input, $u(t) \in \mathcal{R}^r$ is the control input, and $x(t) \in \mathcal{R}^n$ is the state of the system. The regulated output is

$$z(t) = Cx(t) + Du(t), \quad t \in [0,T] \tag{2.3}$$

where $z(t) \in \mathcal{R}^q$. The measurements are sampled:

$$y(ih) = Hx(ih) + v(ih), \quad ih \in (0,T) \tag{2.4}$$

where $v(ih) \in \mathcal{R}^p$ is the (discrete-time) measurement noise. The matrices A, B, E, C, D and H are of compatible dimensions. In this paper, we will assume that $D'D$ is nonsingular.

The $\mathcal{H}_\infty$ control problem we address in this paper is that of designing a *causal* digital controller $\mathcal{K}$ which uses the sampled measurements $y(ih)$ and produces $u(ih)$. The control input is given by

$$u(t) = u(ih), \quad t \in [ih, (i+1)h)$$

In this paper, the performance of a given controller is measured by a certain worst case $\mathcal{H}_\infty$ like performance measure. Recall first that the $\mathcal{H}_\infty$ norm of a linear time-invariant system is the induced norm of corresponding input-output operator obtained by taking $\mathcal{L}_2$(or ℓ_2) norm on the input and output signals. Following the formulation in [3, 11, 18], we introduce the worst case closed loop performance measure $J(G, \mathcal{K})$ defined as

$$J(G, \mathcal{K}) := \sup_{\|w\|^2_{[0,T]} + \|v\|^2_{(0,T)} \neq 0} \left\{ \left[\frac{\|z\|^2_{[0,T]}}{\|w\|^2_{[0,T]} + \|v\|^2_{(0,T)}} \right]^{1/2} \right\}.$$

The performance measure $J(G, \mathcal{K})$ can also be viewed as the *induced norm* of the linear operator generated by the closed loop system which maps (w, v) to z. More explicitly, consider the operator

$$\Gamma : \mathcal{L}_2^m(0, T) \oplus \ell_2^p[0, T] \to \mathcal{L}_2^q[0, T] : \ (w, v) \mapsto z$$

as defined through (2.1, 2.4, 2.3) and the control law $u = \mathcal{K}(y)$. Define the norm in the domain of Γ to be

$$\|(w, v)\|_T := (\|w\|^2_{[0,T]} + \|v\|^2_{(0,T)})^{1/2}.$$

Then the performance measure $J(G, \mathcal{K})$ is the induced norm of the operator Γ.

Our task is to design a causal controller $\mathcal{K}$ which minimizes the worst case performance measures $J(G, \mathcal{K})$. Specifically, we solve the following suboptimal problem:

Given a real number $\gamma > 0$, find, if one exists, a controller $\mathcal{K}$ such that

$$J(G, \mathcal{K}) < \gamma?$$

Once this suboptimal problem is solved, one can then come as close to the optimal performance as desired by iterating on γ.

First we will present a result for the case which is referred to as the *full information* case. More precisely, *full information* means that the controller has access to both $x(t)$ and $w(t)$ for all time. However, the control input is restricted to be piecewise constant.

Lemma 2.1 *Consider the system described by (2.1–2.3). There exists a causal full information controller $\mathcal{K}$ such that $J(G, \mathcal{K}) < \gamma$ if and only if there exists a bounded symmetric function $P(t) \geq 0, t \in [0, T]$ which satisfies the following Riccati differential equation with jumps:*

$$\begin{aligned} -\dot{P}(t) &= A'P + PA + \gamma^{-2}PBB'P + C'C, \quad t \neq ih \\ P(T) &= 0 \\ P(ih^-) &= P(ih) - L(ih)M(ih)^{-1}L(ih)' \end{aligned} \tag{2.5}$$

and

$$\bar{P}(t) := \begin{pmatrix} P(t) & L(t) \\ L(t)' & M(t) \end{pmatrix} \geq 0, M(t) > 0, \quad t \in [ih, (i+1)h)$$

where

$$L(t) := \int_t^{(i+1)h} \Phi_{A+\gamma^{-2}BB'P}(s, t)'(PE + C'D)(s)ds, \quad t \in [ih, (i+1)h) \tag{2.6}$$

$$M(t) := \int_t^{(i+1)h} (E'L + L'E + \gamma^{-2}L'BB'L + D'D)(s)ds, \quad t \in [ih, (i+1)h). \tag{2.7}$$

Moreover, if the above conditions hold, then the control law

$$\mathcal{K}_P : \ u(t) = u(ih) := -M^{-1}(ih)L'(ih)x(ih), \quad t \in [ih, (i+1)h)$$

achieves $J(G, \mathcal{K}_P) < \gamma$.

This lemma shows that even though the controller has access to full information, it only uses the values of the state at the sampling instants. This result is similar to the standard continuous-time case [6], but is in significant contrast to the discrete time case where the controller uses both the x and w in the full information situation. A result quite similar to the above has also been independently obtained in [22] using a different method.

Theorem 2.2 *Consider the system described by (2.1–2.4). There exists a causal controller $\mathcal{K}$ such that $J(G,\mathcal{K}) < \gamma$ if and only if the following conditions hold:*

1. *The matrix function $P(t)$ satisfying the conditions stated in Lemma 2.1 exists.*
2. *There exists a bounded symmetric function $Q(t) \geq 0, t \in [0,T]$ which satisfies the following differential equation with jumps:*

$$\begin{aligned} \dot{Q}(t) &= AQ + QA' + \gamma^{-2}QC'CQ + BB', \quad t \neq ih \\ Q(0) &= 0 \\ Q(ih) &= Q(ih^-)(I + H'HQ(ih^-))^{-1}. \end{aligned} \tag{2.8}$$

3. $\rho(P(t)Q(t)) < \gamma^2, \quad t \in [0,T].$

Moreover, if these conditions hold, then the controller $\mathcal{K}_{PQ}$ as given by

$$\begin{aligned} \dot{\hat{x}}(t) &= (A + \gamma^{-2}BB'P)\hat{x} + (E + \gamma^{-2}BB'L)u, \quad t \neq ih \\ \hat{x}(0) &= 0 \\ \hat{x}(ih) &= \hat{x}(ih^-) + R(ih^-)H'(I + HR(ih^-)H')^{-1}(y(ih) - H\hat{x}(ih^-)) \\ u(t) &= -M^{-1}(ih)L'(ih)\hat{x}(ih), \quad t \in [ih,(i+1)h) \end{aligned} \tag{2.9}$$

where $R(t) := Q(t)(I - \gamma^{-2}P(t)Q(t))^{-1}$, achieves $J(G,\mathcal{K}_{PQ}) < \gamma$.

The condition 1 is necessary as stated in Lemma 2.1 for the existence of any zero order hold controller. The condition 2 is necessary for the existence of a controller when sampled measurements (2.4) are available [18]. The condition 3 is the well known spectral radius condition which occurs in other output feedback problems. In fact, we can combine conditions 2 and 3 into one. We state this result as a corollary.

Corollary 2.3 *Consider the system described by (2.1–2.4). There exists a causal controller $\mathcal{K}$ such that $J(G,\mathcal{K}) < \gamma$ if and only if the following conditions hold:*

1. *The matrix function $P(t)$ satisfying the conditions stated in Lemma 2.1 exists.*
2. *There exists a bounded symmetric function $R(t) \geq 0, t \in [0,T]$ which is the solution of the following differential equation with jumps:*

$$\begin{aligned} \dot{R}(t) &= (A + \gamma^{-2}BB'P)R + R(A + \gamma^{-2}BB'P)' + BB', \quad t \neq ih \\ R(0) &= 0 \\ R(ih) &= R(ih^-)(I + H'HR(ih^-) - \gamma^{-2}LM^{-1}L'(ih)R(ih^-))^{-1}. \end{aligned} \tag{2.10}$$

Moreover, if these conditions hold, then the controller $\mathcal{K}_{PQ}$ as given in Theorem 2.2 achieves $J(G,\mathcal{K}_{PQ}) < \gamma$.

Remarks:

1. The controller described above is a linear time-varying system with jumps. It has the structure of an $\mathcal{H}_\infty$ filter combined with the zero order hold full information control law. The filter is similar to the one given in [18]. In this sense, this controller shares many of characteristics of the central solution to the standard $\mathcal{H}_\infty$ control problem.

2. The Riccati equation for P is similar to the one for Q as in [18] with different jump characterization at jump instants. The Riccati equation for R is in fact one of the two Riccati equations used in the discrete time $\mathcal{H}_\infty$ control problem (see, for example, [3].) Indeed with $R_i := R(ih^-)$, we have:

$$R_{i+1} = \Phi_i R_i (I + H'HR_i - \gamma^{-2} LM^{-1}L'(ih)R_i)^{-1}\Phi_i' + \tilde{B}_i\tilde{B}_i'$$

where $\Phi_i := \Phi_{A+\gamma^{-2}BB'P}((i+1)h, ih)$ and

$$\tilde{B}_i\tilde{B}_i' = \int_{ih}^{(i+1)h} \Phi_{A+\gamma^{-2}BB'P}((i+1)h, s)BB'\Phi_{A+\gamma^{-2}BB'P}((i+1)h, s)'ds.$$

Notice that for the construction of controller only $R_i := R(ih^-)$ is needed.

3. The controller as given in (2.9) can be digitally implemented as following:
Let $\hat{x}_i := \hat{x}(ih^-), u_i := u(ih)$, then a discrete time implemented controller has the form

$$\begin{aligned} \hat{x}_{i+1} &= \Pi_i\hat{x}_i + \Delta_i y(ih) \\ \hat{x}_1 &= 0 \\ u_i &= \Theta_i\hat{x}_i + \Lambda_i y(ih) \end{aligned} \tag{2.11}$$

where

$$\begin{aligned} \Pi_i &:= [\Phi_i - \int_{ih}^{(i+1)h} \Phi_{A+\gamma^{-2}BB'P}((i+1)h, s)(E + \gamma^{-2}BB'P)(s)dsM^{-1}(ih)L'(ih))] \\ &\quad (I + R_iH'H)^{-1} \\ \Delta_i &:= \Pi_iR_iH' \\ \Theta_i &:= -M^{-1}(ih)L'(ih)(I + R_iH'H)^{-1} \\ \Lambda_i &:= \Theta_iR_iH'. \end{aligned}$$

3 Conclusions

In this paper, we have given a solution to the finite horizon $\mathcal{H}_\infty$ control problem for sampled data systems with the restriction of a zero order hold function. The controller has the structure of an $\mathcal{H}_\infty$ estimator for the full information control law in the presence of the worst disturbance. These results show that problem can be decomposed into two parts, namely, full information control and sampled data filtering. The results in this paper are stated for time invariant systems for the sake of simplicity. These results hold as stated for time varying systems as well. The results can also be extended to the case of generalized hold functions easily. Current work is underway to extend the results in this paper to the much more interesting infinite horizon case.

References

[1] B. Bamieh and J. B. Pearson, "A General Framework for Linear Periodic Systems with Application to $\mathcal{H}_\infty$ Sampled–Data Control," Technical report no. 9021, Dept. of Elec. Comp. Eng., Rice University, Nov. 1990.

[2] T. Başar, "Optimal $\mathcal{H}_\infty$ Designs Under Sampled State Measurements," *Syst. Control Lett.*, vol. 16, pp. 399–410, June 1991.

[3] T. Başar and P. Bernhard, *$\mathcal{H}_\infty$ Optimal Control and Related Minimax Design Problems: A Dynamic Game Approach,* Birkhäuser, March 1991.

[4] P. Bernhard, "Application of the Min-max Certainty Equivalence Principle to the Sampled Data Output Feedback $\mathcal{H}_\infty$ Control Problem," *Syst. Control Lett.*, vol. 16, pp. 229–234, April 1991.

[5] T. Chen and B. A. Francis, "Sampled-Data Optimal Design and Robust Stabilization," *Proc. Amer. Control Conf.*, pp. 2704–2709, 1991.

[6] J. C. Doyle, K. Glover, P. P. Khargonekar, and B. A. Francis, "State-space solutions to standard $\mathcal{H}_2$ and $\mathcal{H}_\infty$ control problems," *IEEE Trans. Automatic Control*, vol. AC-34, pp. 831-847, 1989.

[7] B. A. Francis, "Sampled-Data Control Systems", Book of Abstracts, *MTNS 91*, pp. 122–127, To appear in the Proc. 1991 International Symp. Math. Theo. Networks and Syst., Kobe, Japan.

[8] S. Hara and P. T. Kabamba, "Worst Case Analysis and Design of Sampled-Data Control Systems," *Proc. 29th Conf. Decision Control*, pp. 202- 203, 1990.

[9] P. Kabamba, "Control of Linear Systems Using Generalized Sampled- Data Hold Functions," *IEEE Trans. Automatic Control*, vol. AC-32, pp. 772- 783, 1987.

[10] J. P. Keller and B. D. O. Anderson, "A New Approach to Discretization of Continuous-Time Controllers," *Proc. Amer. Control Conf.*, pp. 1133-1136, 1990.

[11] P. P. Khargonekar, K. M. Nagpal and K. Poolla, " $\mathcal{H}_\infty$ Control with Transients," To appear in *SIAM J. Control Optimiz.* 1991.

[12] P. P. Khargonekar and N. Sivashankar, "$\mathcal{H}_2$ Optimal Control for Sampled-Data Systems," To appear in *Proc. 30th Conf. Decision Control*, 1991.

[13] D. Limebeer, B. D. O. Anderson, P. P. Khargonekar and M. Green, "A Game Theoretic Approach to $\mathcal{H}_\infty$ Control for Time Varying Systems," To appear in *SIAM J. Control Optimiz.*, 1991.

[14] B. P. Molinari, "Nonnegativity of a Quadratic Functional," *SIAM J. Control Optimiz.*, vol. 13, pp. 792–806,1975.

[15] K. M. Nagpal and P. P. Khargonekar, "Filtering and Smoothing in an $\mathcal{H}_\infty$ Setting," *IEEE Trans. Automatic Control*, vol. 36, pp. 152-166, 1991.

[16] N. Sivashankar and P. P. Khargonekar, "Robust Stability Analysis of Sampled-Data Systems," To appear in *Proc. 30th Conf. Decision and Control*, 1991.

[17] A. A. Stoorvogel, *The $\mathcal{H}_\infty$ Control Problem: A State-Space Approach,* Ph. D. Thesis, Eindhoven University of Technology, The Netherlands, 1990.

[18] W. Sun, K. Nagpal and P. P. Khargonekar, " $\mathcal{H}_\infty$ Control and Filtering with Sampled Measurements," *Proc. Amer. Control Conf.*, pp. 1652– 1657, 1991.

[19] G. Tadmor, "Worst-Case Design in the Time Domain: The Maximum Principle and the Standard $\mathcal{H}_\infty$ Problem," *Math. Control, Signals, Syst.*, vol. 3, pp. 301–324, 1990.

[20] G. Tadmor, "Optimal $\mathcal{H}_\infty$ Sampled-Data Control in Continuous Time Systems," *Proc. Amer. Control Conf.*, pp. 1658–1663, 1991.

[21] H. T. Toivonen, "Sampled-Data Control of Continuous-Time System with an $\mathcal{H}_\infty$ Optimality Criterion," *Process Control Lab. Report No. 90-1*, Dept. Chem. Eng., Åbo Akademi, Finland, 1990.

[22] H. T. Toivonen, "Sampled-Data $\mathcal{H}_\infty$ Optimal Control of Time-Varying Systems," *Process Control Lab. Report No. 91-7*, Dept. Chem. Eng., Åbo Akademi, Finland, 1990.

[23] Y. Yamamoto, "New Approach to Sampled-Data Control Systems–A Function Space Method," *Proc. 29th Conf. Decision and Control*, pp. 1882–1887, 1990.

inverses,"

Approximately Optimal Finite Dimensional Controllers in the 2-Block H^∞ Problem for Distributed Plants

Hitay Özbay
Department of Electrical Engineering
Ohio State University
Columbus, OH 43210

Abstract

We consider an H^∞ mixed sensitivity minimization problem, for a class of infinite dimensional systems. The optimal controllers for such distributed plants are infinite dimensional. A finite dimensional controller can be obtained by approximating the infinite dimensional parts of the optimal controller. We show that, under certain mild conditions, such a controller stabilizes the original distributed plant and yields a performance arbitrarily close to the optimum. An explicit bound for the performance deviation is given in terms of the approximation errors for the infinite dimensional parts of the optimum controller.

Notation

$\mathbf{R}$: Real numbers,
$\mathbf{C}$: Complex numbers,
RHP: open right half plane in $\mathbf{C}$, $\{s \in \mathbf{C} \ : \ \mathrm{Re}\ s > 0\}$,
D: open unit disc, $\{z \in \mathbf{C} \ : \ |z| < 1\}$,
$j\mathbf{R}$: imaginary axis, $\{s \in \mathbf{C} \ : \ \mathrm{Re}\ s = 0\}$,
$j\mathbf{R}_e$ extended imaginary axis: $\{j\omega \ : \ \omega \in \mathbf{R} \cup \{\infty\}\}$,
$\mathbf{T}$: unit circle, $\{\zeta \in \mathbf{C} \ : \ |\zeta| = 1\}$,
L^∞: Banach space of essentially bounded functions on $j\mathbf{R}$,
H^∞: L^∞ functions which admit analytical extensions to RHP,
L^2: Hilbert space of square integrable functions on $j\mathbf{R}$,
H^2: L^2 functions which admit analytical extensions to RHP,
H^1: Banach space of absolutely integrable functions on $j\mathbf{R}$ which admit analytical extensions to RHP,
$L^\infty(\mathbf{T}), H^\infty(D), L^2(\mathbf{T}), H^2(D), H^1(D)$: replace $j\mathbf{R}$ with $\mathbf{T}$ and RHP with D in the above definitions of L^∞, H^∞, H^2, and H^1, respectively,
$\|G\|_n$: norm of G, when $G \in L^n$, H^n, $L^n(\mathbf{T})$, or $H^n(D)$, $n = 1, 2, \infty$.

1 Introduction

In this paper we consider a 2-block H^∞ mixed sensitivity minimization problem for a class of SISO distributed plants, with finitely many unstable modes. The optimal performance and the optimal controller can be computed from the formulae given in [1], [13], [14]. It is well known that, if the plant to be controlled is infinite dimensional, then the optimal controller is infinite dimensional, too. In this paper we deal with the following question: How can we obtain a finite dimensional controller such that (i) it stabilizes the closed loop system with the original distributed plant, and (ii) the performance of

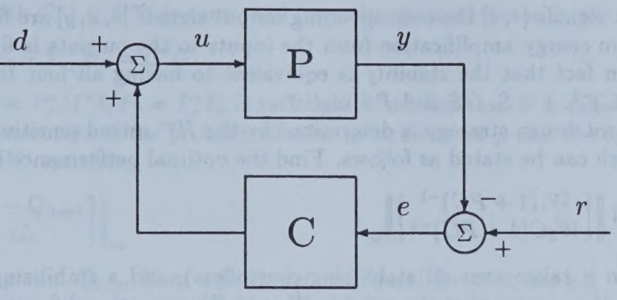


Figure 1: Standard feedback configuration

this design is "close" to the optimum performance level? Such a controller will be called *approximately optimal.* We will obtain an approximately optimal controller by making use of the previous results about the optimal controller and the performance. More specifically we will approximate the *infinite dimensional parts* of the optimal controller to obtain a finite dimensional controller. Then we will show that, under certain mild conditions, this method leads to a finite dimensional approximately optimal controller.

In order to answer the second part of the above question one must be able to compute (or estimate within a specified tolerance) the optimal performance for the infinite dimensional plant. This essentially requires solving the H^∞ problem for distributed plants (which can be done, see e.g. [13]). One can avoid dealing with the H^∞ problem for infinite dimensional systems by first approximating the plant by a finite dimensional system and then estimating the optimal performance for this new plant using the standard state space techniques. (Hoping that the optimal performance, of the new plant, converges to the optimal performance of the original plant, as the new plant converges to the original one.) See [16] and [17] for a discussion on the conditions under which this approach works.

The rest of the paper is organized as follows. In the next section we define the control problem to be considered, and give some preliminary results. In Section 3 we introduce our finite dimensional controller C_f, which is obtained from approximations of the infinite dimensional parts of C_{opt} (the infinite dimensional optimal controller). We show that under certain "mild" conditions C_f approximates C_{opt} in the so-called gap metric. The stability of the closed loop system (C_f, P) is discussed in Section 4. Also in Section 4 we obtain our performance estimate. Finally, we make some concluding remarks in the last section. The interested reader is refered to [12] for the details of the method presented here and an example discussing the numerical issues.

2 Problem definition and preliminary remarks

Consider the feedback control system shown in Figure 1. Here the signals r, d, represent the reference input and the disturbance, respectively. Internal signals in the system are the measured error e, the command signal u, and the output y. The system transfer functions are P, the plant model, and C, the controller to be designed.

We say that the closed loop system (C, P), shown in Figure 1, is *stable* if for all finite energy input signals $[r, d]$ the corresponding output signals $[e, u, y]$ are finite energy, and the maximum energy amplification from the inputs to the outputs is finite. Then, it is a well known fact that the stability is equivalent to having all four transfer functions $S := (1 + PC)^{-1}$, $1 - S$, CS and PS in H^∞.

Our control design strategy is determined by the H^∞ mixed sensitivity minimization criteria, which can be stated as follows. Find the optimal performance level

$$\mu = \inf_C \left\| \begin{bmatrix} W_1(1+PC)^{-1} \\ W_2C(1+PC)^{-1} \end{bmatrix} \right\|_\infty, \tag{1}$$

(the infimum is taken over all stabilizing controllers), and a stabilizing controller C_{opt} achieving μ. We assume that the weights W_1 and W_2 are rational functions in H^∞, with $W_1^{-1}, W_2^{-1} \in H^\infty$.

Given the plant P, all stabilizing controllers are characterized as follows. First write P as a ratio of two strongly coprime functions in H^∞, i.e. find two functions $P_n, P_d \in H^\infty$ such that $P = P_n/P_d$ and

$$\inf_{\mathrm{Re}\, s>0} (|P_n(s)| + |P_d(s)|) > 0.$$

Here we assume that the plant does not have any imaginary axis poles, and has finitely many unstable modes. So, P_d is a rational inner (all-pass) function. Furthermore we assume that the numerator of the plant has an inner outer factorization of the form $P_n = MP_o$ where $M \in H^\infty$ is inner (possibly infinite dimensional) and P_o is rational and outer (minimum phase). A typical example is finite dimensional systems with time delays.

Then, we can find X and Y in H^∞, solving the Bezout equation

$$P_n(s)X(s) + P_d(s)Y(s) = 1. \tag{2}$$

Since P_d is rational there are finitely many interpolation conditions on $X(s)$ for $Y = (1 - XP_n)/P_d$ to be in H^∞. So, X can be chosen as a rational function. A controller stabilizes the plant P if and only if it is of the form

$$C = \frac{X + P_dQ}{Y - P_nQ} \tag{3}$$

where $Q \in H^\infty$ is the free parameter to be determined from the control problem (1). Using the parametrization (3) in (1) we can show that μ satisfies

$$\mu = \inf_{Q \in H^\infty} \left\| \begin{bmatrix} W_1 \\ 0 \end{bmatrix} - \begin{bmatrix} W_1P_n \\ -W_2P_d \end{bmatrix} (X + P_dQ) \right\|_\infty .$$

This problem can further be simplified to (see e.g. [2])

$$\mu = \inf_{Q_0 \in H^\infty} \left\| \begin{bmatrix} W_0 - Q_0 \\ G_0 \end{bmatrix} \right\|_\infty, \tag{4}$$

where

$$W_0 := (P_d^* P_n^* G^{-*} W_1^* W_1 - P_d^* X G) \qquad Q_0 := GQ,$$

$$G_0 := W_1 W_2 G^{-1},$$

and $G \in H^\infty$, with $G^{-1} \in H^\infty$, is computed from the spectral factorization

$$W_1^* P_n^* P_n W_1 + W_2^* W_2 = G^* G. \tag{5}$$

Note that $P_n^* P_n = P_o^* M^* M P_o = P_o^* P_o$ is rational, so the function G is rational.

This special structure of the problem enables us to solve for μ and the optimal interpolant $Q_{0,opt} \in H^\infty$ satisfying

$$\mu = \left\| \begin{bmatrix} W_0 - Q_{0,opt} \\ G_0 \end{bmatrix} \right\|_\infty . \tag{6}$$

One can obtain μ and $Q_{0,opt}$ from a *finite determinantal formula*, see e.g. [13] and [14] for details. We can then compute the optimal controller

$$C_{opt} = \frac{X + P_d Q_{opt}}{Y - P_n Q_{opt}} \quad \text{where} \quad Q_{opt} = G^{-1} Q_{0,opt}. \tag{7}$$

We deduce from (7) and (2) that whenever the inner part M of the numerator, P_n, of the plant is infinite dimensional, the optimal controller C_{opt} is infinite dimensional, too. Therefore, we are interested in finding an appropriate finite dimensional, say kth order, approximation C_k for C_{opt}, in the following sense. The controller C_k must stabilize the distributed plant P, i.e. the closed loop system (C_k, P) should be stable. Also, for any performance tolerance $\varepsilon > 0$ we want to find an integer k sufficiently large, such that the performance of (C_k, P), i.e.

$$\mu_k := \left\| \begin{bmatrix} W_1 (1 + P C_k)^{-1} \\ W_2 C_k (1 + P C_k)^{-1} \end{bmatrix} \right\|_\infty , \tag{8}$$

is within ε of the optimal performance, $|\mu_k - \mu| \leq \varepsilon$. That is, we want that $\mu_k \to \mu$, as $k \to \infty$.

In the next section we propose an approximation scheme to compute such a finite dimensional controller, from the structure (7) of the optimal controller. The topology in which we make our approximations is known as the gap metric, (see [3] for a precise definition and further references).

3 Finite dimensional controllers via approximations in the gap metric

Recall the structure of the optimal controller from (7) and note that X, and P_d are finite dimensional and Y, P_n, Q_{opt} are infinite dimensional. Therefore, a finite dimensional controller can be obtained by simply replacing Y, P_n, Q_{opt} by their kth order, finite dimensional, "approximates" $Y_k, P_{n,k}, Q_k$:

$$C_k := \frac{X + P_d Q_k}{Y_k - P_{n,k} Q_k}, \tag{9}$$

(here by a slight abuse of notation we write C_k for the finite dimensional controller (9) whose order is larger than k). We want to study the conditions under which one can obtain appropriate $Y_k, P_{n,k}, Q_k$ so that the controller C_k is "close" to the optimal controller C_{opt} in the gap metric. Let us re-write C_k as

$$C_k = \frac{(X + P_d Q_{opt}) + \Delta_k^n}{(Y - P_n Q_{opt}) + \Delta_k^d}, \tag{10}$$

where

$$\Delta_k^n = P_d(Q_k - Q_{opt}) \tag{11}$$

and

$$\Delta_k^d = (Y_k - Y) - P_{n,k}(Q_k - Q_{opt}) - Q_{opt}(P_{n,k} - P_n). \tag{12}$$

Set $C_{opt}^n := (X + P_d Q_{opt})$, $C_{opt}^d := (Y - P_n Q_{opt})$, and define their spectral factor $G_c, G_c^{-1} \in H^\infty$

$$G_c^* G_c := C_{opt}^{n*} C_{opt}^n + C_{opt}^{d*} C_{opt}^d. \tag{13}$$

The fact that $G_c^{-1} \in H^\infty$ comes from strong coprimeness of C_{opt}^n and C_{opt}^d, see [18]. We then conclude that the following two statements are equivalent, (see Lemmas 2 and 3 of [3]).

$$(a): \quad \left\| G_c^{-1} \begin{bmatrix} \Delta_k^n \\ \Delta_k^d \end{bmatrix} \right\|_\infty \longrightarrow 0 \quad \text{as} \quad k \to \infty, \tag{14}$$

and (b): finite dimensional controllers C_k, $k \geq 1$, converge to the optimal controller C_{opt}, in the *gap metric*, as $k \to \infty$.

Since $G_c^{-1} \in H^\infty$, (14) holds if and only if the functions Y, P_n and Q_{opt} are uniformly approximable in H^∞ by rational functions. On the other hand any function $F \in H^\infty$ is uniformly approximable by a rational function if and only if its boundary value function $F(j\omega)$ is continuous on the extended imaginary axis $j\mathbf{R}_e$, see for example [8] and [9]. Accordingly, we assume that $P_n(j\omega)$ is continuous on $j\mathbf{R}_e$. This guarantees that $Y(j\omega)$ is continuous on $j\mathbf{R}_e$, because $Y = (1 - P_n X)/P_d$ and, X and P_d are rational. Therefore, we need to find conditions under which $Q_{opt}(j\omega)$ is continuous on $j\mathbf{R}_e$.

We transform the data to the unit disc, from the right half plane, by using a conformal map $z = \frac{s-1}{s+1}$, $s = \frac{1+z}{1-z}$, Re $s \geq 0$, and $|z| \leq 1$. The function $q_{opt}(z) := Q_{opt}(s)|_{s=\frac{1+z}{1-z}}$, is analytic in D, and inherits all the properties of the function $Q_{opt}(s)$ defined in RHP. For example, $\|q_{opt}\|_\infty = \|Q_{opt}\|_\infty$, and the boundary value function $Q_{opt}(j\omega)$ is continuous on $j\mathbf{R}_e$ if and only if $q_{opt}(\zeta)$ is continuous on the unit circle $\mathbf{T}$. Also one can easily go back, from the unit disc, to the right half plane by the inverse transformation $Q_{opt}(s) = q_{opt}(z)|_{z=\frac{s-1}{s+1}}$. The uniform rational approximations of $Q_{opt}(s)$ can be obtained from the uniform rational approximations of $q_{opt}(z)$, for example see [8] for the details of an FFT type algorithm. Recall that $Q_{opt} = G^{-1} Q_{0,opt}$, where G defined by (5) is rational with $G^{-1} \in H^\infty$, and $Q_{0,opt}$ is obtained from (6). So, if $q_{0,opt}(\zeta)$, $|\zeta| = 1$, is continuous then $Q_{opt}(j\omega)$ is continuous, $j\omega \in j\mathbf{R}_e$. Hence, we want to understand the continuity properties of $q_{0,opt}$.

It is well known, see e.g. [2], that the optimal interpolant $Q_{0,opt}$ of (6) satisfies

$$|W_0(j\omega) - Q_{0,opt}(j\omega)|^2 + |G_0(j\omega)|^2 = \mu^2.$$

We know that $\mu \geq \|G_0\|_\infty$. In fact, when $\mu = \|G_0\|_\infty$, it is not possible to obtain a straightforward formula for the optimal controller of the original 2-block problem (4). Therefore, we assume that $\mu > \|G_0\|_\infty$. In which case we can find a rational function $L \in H^\infty$ with $L^{-1} \in H^\infty$ such that $|L(j\omega)|^2 = \mu^2 - |G_0(j\omega)|^2$. Hence $Q_{L,opt} := L^{-1} Q_{0,opt} \in H^\infty$ is the optimal interpolant for the 1-block problem [2]

$$\inf_{Q_L \in H^\infty} \|W_0/L - Q_L\|_\infty =: \|W_0/L - Q_{L,opt}\|_\infty = 1. \tag{15}$$

We know, from the arguments above that when $P_n(j\omega)$ is continuous on $j\mathbf{R}_e$ the function $W_0(j\omega)/L(j\omega)$ is continuous on $j\mathbf{R}_e$. However, this is not sufficient for $Q_{L,opt}(j\omega)$ (hence for $Q_{0,opt}(j\omega)$ and $Q_{opt}(j\omega)$) to be continuous on the extended imaginary axis, (see [15], p. 48). Therefore, we need some more conditions on the continuous L^∞ function $W_0/L =: \widehat{W}$. Let us consider $\widehat{w}(\zeta)$, $|\zeta| = 1$, the transformed version of $\widehat{W}(j\omega)$, $j\omega \in j\mathbf{R}_e$. Since $\widehat{W} \in L^\infty$, we have that $\widehat{w} \in L^\infty(\mathbf{T})$ with a Fourier series expansion

$$\widehat{w}(\zeta) =: \sum_{\ell=-\infty}^{\infty} \widehat{w}_\ell \, \zeta^\ell . \tag{16}$$

Note also that $\widehat{w}$ is continuous on $\mathbf{T}$. The following result is a special case of Peller-Hruscev theorem.

Lemma 3.1: Notation as above, if

$$\sum_{\ell=-\infty}^{\infty} |\widehat{w}_\ell| < \infty, \tag{17}$$

then the optimal interpolant $q_{L,opt}(z)$, $|z| \leq 1$, of the 1-block problem (15) has a power series expansion

$$q_{L,opt}(z) = \sum_{\ell=0}^{\infty} (q_{L,opt})_\ell \, z^\ell, \quad \text{with} \quad \sum_{\ell=0}^{\infty} |(q_{L,opt})_\ell| < \infty,$$

and $q_{L,opt}(\zeta)$, hence $Q_{L,opt}(j\omega)$ and $Q_{opt}(j\omega)$, are continuous, $\zeta \in \mathbf{T}$ and $j\omega \in j\mathbf{R}_e$.
Proof: See [15] pp. 48–53. □
The above lemma gives us a sufficient condition for uniform approximability, in the gap metric, of the optimal controller C_{opt}, by the rational functions C_k, (10). This condition is expressed in terms of the absolute convergence of the Fourier series of $\widehat{w}(\zeta)$. In terms of the original problem data $P = P_n/P_d$, and W_1, W_2, (17) can be expressed as follows.

Lemma 3.2: Consider the numerator part of the plant $P_n \in H^\infty$ with $P_n(j\omega)$ continuous on $j\mathbf{R}_e$. Let P_n be in H^2 and further assume that $\frac{d}{ds}P_n \in H^1$. Then $Q_{opt}(j\omega)$ is continuous on $j\mathbf{R}_e$, and moreover $q_{opt}(z)$ has absolutely convergent power series expansion, for $|z| \leq 1$

$$q_{opt}(z) = \sum_{\ell=0}^{\infty} (q_{opt})_\ell \, z^\ell \quad \text{and} \quad \sum_{\ell=0}^{\infty} |(q_{opt})_\ell| < \infty. \tag{18}$$

Thus q_{opt} is uniformly approximable by polynomial functions (of z)

$$q_k(z) = \sum_{\ell=0}^{k} (q_{opt})_\ell \, z^\ell, \tag{19}$$

i.e $\|q_{opt} - q_k\|_\infty \to 0$ as $k \to \infty$. This means that Q_{opt} is uniformly approximable by rational functions $Q_k(s) = q_k(z)|_{z=\frac{s-1}{s+1}}$.

Proof: We know, from Theorem 2.12 of [8], that if $P_n(s)$ satisfies the hypotheses of the lemma then $p_n(z) = P_n(s)|_{s=\frac{1+z}{1-z}}$ has an absolutely convergent power series, $|z| \leq 1$. Therefore, since $\widehat{w}$ is defined in terms of multiplications and additions of rational functions with p_n, we have that $\widehat{w}$ is in the Wiener algebra of continuous functions (on $\mathbf{T}$) with absolutely convergent Fourier series. Hence the result follows from Lemma 3.1. □

We refer the interested reader to [8] for a discussion on the assumptions used in Lemma 3.2, and convergence properties of the approximations. Let us summarize the above discussion with the following.

Theorem 3.3: Consider the 2-block H^∞ mixed sensitivity minimization problem (1), with the plant $P = MP_o/P_d$ and the weights W_1 and W_2, where P_o, P_d, W_1, W_2 are rational (with P_d inner P_o outer, $W_1^{-1}, W_2^{-1} \in H^\infty$) and M is arbitrary inner. Assume that $P_n = MP_o$ is continuous on $j\mathbf{R}_e$, $P_n \in H^2$, and $\frac{d}{ds}P_n \in H^1$. Then the optimal controller C_{opt} can be uniformly approximated, in the gap metric, by rational functions C_k of the form (10):

$$C_k = \frac{X + P_d Q_k}{Y_k - P_{n,k} Q_k},$$

where the rational function $X \in H^\infty$ is determined from (2), Y_k, $P_{n,k}$ and Q_k are rational (kth order) approximations of Y, P_n and Q_{opt} respectively, such that

$$\|Y - Y_k\|_\infty \to 0 \quad \text{as} \quad k \to \infty,$$

$$\|P_n - P_{n,k}\|_\infty \to 0 \quad \text{as} \quad k \to \infty,$$

$$\|Q_{opt} - Q_k\|_\infty \to 0 \quad \text{as} \quad k \to \infty.$$

Proof: Immediate from the above discussion and Lemmas 3.1 and 3.2. □

The approximation procedure summarized by (19) in Lemma 3.2, can be implemented using an FFT type algorithm of [8]. This is just one way to obtain finite dimensional approximations of infinite dimensional systems. There are several other approximation methods available, see for example [5] [10] [11], [6] [19] and references in these papers. However, the DFT/FFT type algorithms are more suitable to implement the controller of [13], [14] where the optimal interpolant Q_{opt} can be easily evaluated at some finitely many points $j\omega_1, j\omega_2, \ldots, j\omega_N$ on the imaginary axis, (or equivalently q_{opt} can be evaluated at points $e^{j\theta_1}, e^{j\theta_2}, \ldots, e^{j\theta_N}$ on the unit circle). On the other hand the closed form expression of Q_{opt} can be rather complicated.

4 Stability and performance of the closed loop system (C_k, P)

In this section we assume that the hypotheses of Theorem 3.3 hold, so that the finite dimensional controller C_k approximates the optimal controller C_{opt} in the gap metric. That is C_k given by (10) satisfies (14). The stability of the closed loop system with the finite dimensional controller C_k and the original distributed plant P is shown in the following.

Theorem 4.1: Assume that the hypotheses of Theorem 3.3 hold. Then there exists a sufficiently large number K such that the closed loop system (C_k, P) is stable for all $k \geq K$.

Proof: Since P is stabilizable and C_{opt} stabilizes P, we know that there is a positive number $b_{P,C_{opt}} > 0$, such that the closed loop system (C_k, P) is stable if

$$\|G_c^{-1} \begin{bmatrix} \Delta_k^n \\ \Delta_k^d \end{bmatrix} \|_\infty < b_{P,C_{opt}}, \tag{20}$$

where G_c, Δ_k^n and Δ_k^d are defined by (13), (11), (12). See [3] [4] for a proof of this fact. On the other hand by Theorem 3.3

$$\varepsilon(k) := \|G_c^{-1} \begin{bmatrix} \Delta_k^n \\ \Delta_k^d \end{bmatrix} \|_\infty \to 0 \quad \text{as} \quad k \to \infty. \square \tag{21}$$

It is important to note that the quantity $b_{P,C_{opt}}$ can be computed from the formula [3]

$$b_{P,C_{opt}}^{-1} = \| \begin{bmatrix} 1 \\ P \end{bmatrix} (1 + PC_{opt})^{-1} [1 \quad C_{opt}] \|_\infty ,$$

which is equal to (see e.g. [7])

$$b_{P,C_{opt}}^{-1} = \left\| G_p G_p^{-*} \begin{bmatrix} (P_n^* Y - P_d^* X) - G_p G_p^* Q_{opt} \\ 1 \end{bmatrix} \right\|_\infty , \tag{22}$$

where $G_p, G_p^{-1} \in H^\infty$, and

$$G_p G_p^* = P_n P_n^* + P_d P_d^* .$$

Therefore, from (22) we can determine how small $\varepsilon(k)$ in (21) should be for stability of the feedback system (C_k, P). This also determines how large K should be in Theorem 4.1.

An important requirement in the approximation scheme C_k is that the performance μ_k of the system (C_k, P), should be close to the optimum μ. The next result shows that C_k, obtained from the approximation method used in Theorem 3.3, satisfies this requirement.

Theorem 4.2: Assume that the hypotheses of Theorem 3.3 hold. Then

$$\left\| \begin{bmatrix} W_1(1 + PC_k)^{-1} \\ W_2 C_k (1 + PC_k)^{-1} \end{bmatrix} \right\|_\infty =: \mu_k \to \mu := \left\| \begin{bmatrix} W_1 (1 + PC_{opt})^{-1} \\ W_2 C_{opt} (1 + PC_{opt})^{-1} \end{bmatrix} \right\|_\infty , \tag{23}$$

as $k \to \infty$.

Proof: First define $C_k^n := X + P_d Q_k$, $C_k^d := Y_k - P_{n,k} Q_k$, and $S_{opt} := (1 + PC_{opt})^{-1}$. Then, note that since (C_{opt}, P) is a stable closed loop system we have

$$G_L := (P_d C_{opt}^d + P_n C_{opt}^n)^{-1} \in H^\infty .$$

In fact the Bezout identity implies that $G_L = 1$. One can re-write μ_k as

$$\mu_k = \left\| \begin{bmatrix} W_1 P_d C_k^d (P_d C_k^d + P_n C_k^n)^{-1} \\ W_2 P_d C_k^n (P_d C_k^d + P_n C_k^n)^{-1} \end{bmatrix} \right\|_\infty .$$

Then, it is easy to see that

$$\mu_k = \left\| \begin{bmatrix} W_1 S_{opt} + W_1 P_d G_L \Delta_k^d \\ W_2 C_{opt} S_{opt} + W_2 P_d G_L \Delta_k^n \end{bmatrix} \left(\frac{1}{1 + G_L (P_d \Delta_k^d + P_n \Delta_k^n)} \right) \right\|_\infty .$$

Recall that P_d is inner and $G_L = 1$, so we have

$$\mu_k \leq \left(\mu + \left\| \begin{bmatrix} W_1 \Delta_k^d \\ W_2 \Delta_k^n \end{bmatrix} \right\|_\infty \right) \left(\frac{1}{1 - \|(P_d \Delta_k^d + P_n \Delta_k^n)\|_\infty} \right) . \tag{24}$$

On the other hand, by Theorem 3.3, $\|\Delta_k^n\|_\infty \to 0$, and $\|\Delta_k^d\|_\infty \to 0$, as $k \to \infty$. Thus we conclude that $\mu_k \to \mu$ as $k \to \infty$. $\square$

The above theorem proves the convergence of μ_k to μ. It also gives an explicit bound (24) for the performance degradation. This bound is expressed in terms of the approximation errors Δ_k^n and Δ_k^d, which depend on the specific method used in the approximations Q_k, Y_k and $P_{n,k}$.

5 Conclusions

In this paper we have considered a 2-block H^∞ problem for a class of possibly unstable infinite dimensional systems. Our main purpose was to obtain a finite dimensional stabilizing controller achieving a performance "close" to the optimum. In order to obtain such a controller we used the method of approximating the infinite dimensional parts of the optimal controller. For this class of systems the H^∞ optimal controller and the performance can be computed from the formulae given in [13], [14]. We showed that, under certain mild conditions on the plant, the optimal controller is uniformly approximable by rational functions in the gap metric topology. We have shown that such finite dimensional controllers yield a stable closed loop system with a performance close to the optimum.

The reason we have considered the so-called S and CS mixed sensitivity minimization problem is that in this problem the optimal interpolant Q_{opt} is a proper function in H^∞, provided $W_1^{-1}, W_2^{-2} \in H^\infty$. So, it is possible to talk about finite dimensional approximations. However, for the so-called S and $(1-S)$ mixed sensitivity minimization problem, the "optimal" interpolant is improper, if the plant is strictly proper and the weights are proper. Therefore, it cannot be approximated *uniformly* (i.e. in $\|\cdot\|_\infty$ norm) by rational functions. Hence the optimal controller cannot be approximated by rational functions, in the gap metric, because the term $P_d Q_{opt}$ in the numerator of C_{opt} is improper. See [17] for a comparison of well-posedness of these, and several other H^∞ optimal control problems.

Finally, we also want to mention that the Theorems 3.3, 4.1 and 4.2 hold for any distributed plant P with P_n satisfying the hypotheses of Lemma 3.2. In particular one does not need the assumption that the outer part of P_n is rational.

References

[1] Flamm, D. and H. Yang, "H^∞ optimal mixed sensitivity for general distributed systems," Proc. of the 29th IEEE Conference on Decision and Control, Honolulu, Hawaii, December 1990, pp. 134–139.

[2] Francis, B. A., *A Course in H^∞ Control Theory*, Lecture Notes in Control and Information Sciences, vol. 88, Springer Verlag, 1987.

[3] Georgiou, T. T. and M. C. Smith, "Optimal robustness in the gap metric," *IEEE Transactions on Automatic Control* **35** (1990), pp. 673–686.

[4] Georgiou, T. T. and M. C. Smith, "Robust control of feedback systems with combined plant and controller uncertainity," Proceedings of the American Control Conference, San Diego CA, May 1990, pp. 2009–2013.

[5] Glover, K., R. F. Curtain and J. R. Partington, "Realisation and approximation of linear infinite dimensional systems with error bounds," *SIAM J. Control and Optimization*, **26** (1988), pp. 863–898.

[6] Glover, K., J. Lam and J. R. Partington, "Rational approximation of a class of infinite dimensional systems, I: singular values of Hankel operators," *Mathematics of Control, Signals and Systems*, **3** (1990), pp. 325–344.

[7] Glover, K. and D. McFarlane, "Robust stabilization of normalized coprime factor descriptions with H^∞ bounded uncertainity," *IEEE Transactions on Automatic Control*, **34** (1989), pp. 821–830.

[8] Gu, G., P. P. Khargonekar and E. B. Lee, "Approximation of infinite dimensional systems," *IEEE Transactions on Automatic Control*, **34** (1989), pp. 610–618.

[9] Hoffman, K., *Banach Spaces of Analytic Functions*, Dover Publications, New York, (reprint, originally published in 1962 by Prentice Hall).

[10] Mäkilä, P., "Approximation of stable systems by Laguerre filters," *Automatica* **26** (1990), pp. 333–345.

[11] Mäkilä, P., "Approximation and identification of continuous time systems," *Int. J. Control*, **52** (1990), pp. 669–687.

[12] Özbay, H., "Controller reduction in the 2-Block H^∞ optimal design for distributed plants," to appear in *Int. J. Control.*

[13] Özbay, H., M. C. Smith and A. Tannenbaum, "Mixed sensitivity optimization for unstable infinite dimensional systems," Technical Report, Dept. of Electrical Engineering, University of Minnesota, December 1989. A short version of the paper, under the title "Controller design for unstable distributed plants," appears in the proceedings of the American Control Conference, San Diego CA, May 1990, pp. 1583–1588.

[14] Özbay, H., M. C. Smith and A. Tannenbaum, "On the optimal two block H^∞ compensators for distributed unstable systems," submitted to 1992 American Control Conference.

[15] Power, S. C., *Hankel Operators on Hilbert Space*, Pitman Advanced Publishing Program, Boston, 1982.

[16] Rodriguez, A. and M. A. Dahleh, "Weighted H^∞ optimization for stable infinite dimensional systems using finite dimensional techniques," Proc. of the 29th IEEE Conference on Decision and Control, Honolulu, Hawaii, December 1990, 1814–1820.

[17] Smith, M. C., "Well-posedness of H^∞ optimal control problems," *SIAM J. Control and Optimization*, **28** (1990), pp. 342–358.

[18] "On stablization and existence of coprime factorizations," *IEEE Transactions on Automatic Control* **34** (1989), pp. 1005–1007.

[19] Wu, N. E. and G. Gu, "Discrete Fourier transform and H^∞ approximation," *IEEE Transactions on Automatic Control* **35** (1990), pp. 1044–1046.

Optimal bi-level quantization *

Jianghai Fu
Dept. of Computer Science
Univ. of Western Ontario
London, Ont. N6A 5B7
CANADA

Shouwen Tang
Dept. of Computer Science
Beijing Computer Institute
Beijing, China

Kaizhong Zhang
Dept. of Computer Science
Univ. of Western Ontario
London, Ont. N6A 5B7
CANADA

Abstract. A 2-clustering of a given set of points in the plane is a partition of the points into 2 subsets (clusters). We present an algorithm which minimizes the mean-square error measure of the clusters in $O(n^2 \log n)$ time and $O(n)$ space. This problem is known as bi-level quantization.

1 Introduction and definitions

Let S be a set of n points in the plane. A partition of S into 2 disjoint sets A_1 and A_2 is called a 2-clustering.

Let W be some weight function that assigns a real weight to any finite set A of points in the plane, like the diameter of A, the radius of A, or the variance of A. Let F be a 2-ary symmetric function, assigning a real value to every 2-tuple of reals.

The 2-clustering problem for W with respect to F is defined as follows: Given a set S of n points in the plane; find the 2-clustering of S into A_1 and A_2 such that $F(W(A_1), W(A_2))$ is minimized. An optimal bi-level quantization is a 2-clustering problem where W is the variance and F is function summation.

More precisely, an optimal bi-level quantization is a partition of S into 2 disjoint sets A_1 and A_2 such that

$$\sum_{a_i \in A_1} (a_i - a_1)^2 + \sum_{a_i \in A_2} (a_i - a_2)^2$$

, where $a_1 = \sum_{a_i \in A_1} a_i / \mid A_1 \mid$ and $a_2 = \sum_{a_i \in A_2} a_i / \mid A_2 \mid$, is minimized over all possible bi-partition of the set S.

The 2-clustering problem for the maximum diameter has been treated by Asano, Bhattacharya, Keil, and Yao [ABKY88]. They gave an $O(n \log n)$ algorithm for this problem. Monma and Suri [MS88] gave an $O(n^2)$ algorithm for finding a 2-clustering with smallest sum of diameters.

Bi-level quantization in $1-D$ has a trivial $O(n \log n)$ algorithm. However the results for $2-D$ and higher dimensions are unknown. In this paper we give an optimal solution for $2-D$ with a time complexity of $O(n^2 \log n)$ and space complexity $O(n)$.

---

*Research supported by the Natural Sciences and Engineering Research Council of Canada grants OGP0046373. Shouwen Tang is currently visiting Department of Computer Science, University of Western Ontario.

Optimal bi-level quantization has application in multilevel quantization. We can use bi-level quantization to perform tree-structured vector quantization. Optimal bi-level quantization is also useful to provide a good initialization of Max's algorithm.

2 Lemmas

Our results are based on the following Lemmas. Without loss of generality, we assume that the the set S is not degenerate, i.e. there are three points in S which are not collinear.

Lemma 1 *The optimal partition A_1 and A_2 can be separated by a line.*

Proof: Trivial. □

Lemma 2 *Let A_1 and A_2 be a partition of S by a line, there is always another line which also partition S into A_1 and A_2 and is determined by two points in set S. Furthermore the two points are both in either set A_1 or set A_2.*

Sketch of the proof: Let the partition line be L We move line L parallelly to A_1 until it meet one point, say p_1, in A_1. Let this new line be L_1. We then move line L parallelly to A_2 until it meet one point, say p_2, in A_2. Let this new line be L_2.

If in L_1 there are more than one points, then they must be all from A_1. In this case, L_1 is the the line we want. Similarly if in L_2 there are more than one points, then they must from A_2. In this case, L_2 is the the line we want.

If p_1 is the only point in L_1 and p_2 is the only point in L_2, we can rotate L_1 around p_1 and L_2 around p_2 simultaneously and still keep them parallel. We will rotate them until either L_1 meet another point in A_1 or L_2 meet another point in A_2. This is always possible since the set S is not degenerate. Now the line that has two points is the line we want. □

Lemma 3 *There are at most n^2 different partitions.*

Proof: From Lemma 2, we know that there are at most $n^2/2$ different partitions lines. For each of the partition line, there are two different partitions since the two points in the line can be either in A_1 or in A_2. Therefore there are at most n^2 different partitions. □.

3 Algorithm

Based on the lemmas, we develop following algorithm.

Step 1, Pick up a point P from S.

Step 2, Use P as the center and fix a direction d. For any point p in S, compute $a =$ the angle between d and line segment from P to p, and $dis =$ the distance from p to P. The dis is considered positive if p is in the first two quadrants and negative if it is in the last two quadrants.

Step 3, Sort all points in S by (a, dis). We sort them first by angle and if two points have same angle use their distance to p to determine the order.

Note each angle appeared in the sorted list represents a line pass through P. Because of Lemma 2 and Lemma 3, we do not need to consider any angle which do not appear in the sorted list.

Step 4. Pick up an angle from the sorted list and compute the mean-square error measure determined by the line of this angle.

Step 5 Rotate the line by scan the sorted list of points. and compute the the mean-square error measure for each angle appeared in the sorted list. In the computation, we find the angle which generate minimum mean-square error measure.

Above steps exhaust all the lines pass through P. The time for step 1 is constant, for step 2 is $O(n)$, for step 3 is $O(n \log n)$, for step 4 is $O(n)$, and for step 5 is $O(n)$. Note that we omit the detail of how to compute the mean-square error for a given angle in step 4 and how to do step 5 in $O(n)$ time.

If we try all the points in S then we can find the optimal solution. Therefore the time complexity will be $O(n^2 \log n)$.

4 Discussion

Our result can be generalized to higher dimensions. In 3-D, we can find the optimal bi-level quantization of n pints in $O(n^3 \log n)$. We can also generalize our result to k-level quantization for fixed k. We proved that for any fixed k, the k-level quantization can be done in polynomial time.

One interesting open problem is if one can improve the time complexity of our algorithm. It is likely that one may achieve $O(n^2)$ time complexity. However, breaking the $O(n^2)$ time complexity barrier seems rather hard.

References

[1] T. Asano, B. Bhattacharya, M. Keil, and F. Yao, 'Clustering algorithms based on minimum and maximum spanning trees' *Proceedings Fourth Annu. Symp. Computational Geometry*, Urbana-Champaign, pp. 252-257, 1988

[2] C. Monma and S. Suri 'Partitioning points and graphs to minimize the maximum or the sum of diameters' *The Theory and Applications of Graphs, Proc. Sixth Int. Conf. Theory and Appl. of Graphs*, Michigan, 1988

[3] V. Capoyleas, G. Rote, and G. Woeginger 'Geometric Clustering', *J. of Algorithms*, vol. 12, pp.341-356, 1991

Methods for Approximating the Short-time Energy of Noise-corrupted Speech Signals

Steven A. Dimino, Russell J. Niederjohn,
James A. Heinen, and Thomas J. Svoren
Department of Electrical and Computer Engineering,
Marquette University, Milwaukee WI 53233

Introduction

Current research suggests that a feature-based speech processing system, where features extracted from speech corrupted by white noise are used to control a cepstral processor, has promise in enhancing the intelligibility of the noise-corrupted speech signal. Presently the features being extracted include: (i) the fundamental frequency (pitch), (ii) estimates of the first three formant frequencies, and (iii) voiced/unvoiced decision (as well as confidence measures for each extracted feature). However, one important feature that until recently has been overlooked is the short-time energy contour. It has been determined [1] that processing consisting only of adjusting the short-time energy contour of the noise-corrupted speech signal to match that of the noise-free short-time energy contour has resulted in MDRT (Modified Diagnostic Rhyme Test) scores that are virtually identical to those obtained using features extracted from a corrupted signal at +6 db above the actual signal-to-noise ratio (SNR). This significant result serves as justification for an in-depth study of possible methods of approximation the uncorrupted short-time energy contour from the noise-corrupted speech signal utilizing statistical characterizations of both the speech signal and corruptive noise.

Discussion

The short-time energy of a rectangularly windowed segment (of length N)of speech can be expressed as

$$E_n = \sum_{m=n-N+1}^{n} s^2(m) . \qquad (1)$$

However, when an additive white noise sequence d(n) has corrupted the discrete-time signal s(n) to produce x(n) = s(n) + d(n), the short-time energy of s(n) given in (1) cannot be computed directly and must be statistically approximated. One possible way to estimate the uncorrupted contour that has been under study is by subtracting the variance of d(n) from the noise-corrupted short-time energy of x(n) calculated in (1). The problem of approximating the short-time energy of s(n) from the short-time energy of x(n) is thus reduced to that of estimating the noise variance on a short-time basis.

Conventionally, noise variance is estimated from the portions of a speech segment where it is known that no speech exists. This method has two drawbacks. The first is that at low SNR's it is difficult to identify silent segments in continuous speech. Secondly, it assumes that the noise is stationary over an indefinite period of time. Two methods of noise variance estimation which attempt to overcome the deficiencies in the conventional method are now described.

The first is Paliwal's method [2] which is based on the AR parameters of a sub-optimal ARMA model [3] of the speech signal. These AR parameters are derived from the high-order Yule-Walker equations and are used in conjunction with the correlation matching property of AR spectral estimators in order to extrapolate an estimate of the lag-zero value of the uncorrupted signals autocorrelation function. This is the desired energy value. The second is a new method that is proposed as an improvement (in terms of robustness) to Paliwal's method. It is based on the AR modeling of the 'time-series' that consists of a particular set of autocorrelation lags (avoiding lag-zero) estimated from the noise-corrupted speech signal. With these AR parameters a one-step backward prediction can be effected to again estimate the desired lag-zero value.

Results

The two methods were compared to the conventional method in terms of their effectiveness in approximating the uncorrupted contour in a mean-square sense. In tests where the noise was generated to be white and wide-sense stationary, Paliwal's method performed best for SNR's above 0 db but its performance was poor for lower SNR's. The conventional method performed relatively well for all SNR's tested. For very low SNR's (-6 to -12 db) the proposed method performed better than either the conventional method or Paliwal's method. Tests involving non-stationary noise also need to be conducted since this test was biased towards the conventional method. Since the effect of the proposed processing on the intelligibility of speech is of interest, MDRT testing must also be conducted. These tests will be conducted in the near future.

Conclusions

Motivation for researching methods of estimating the short-time energy contour of speech based on the encouraging MDRT testing from [1] was presented. Two previously studied methods (the conventional method and Paliwal's method), as well as a new method for estimating the short-time energy contour of speech were described. Objective mean-square-error tests were conducted and showed that the proposed method was an improvement on the first two. Finally, it was noted that more intelligibility testing needs to be done in order to fully understand the role of the short-time energy contour in intelligibility enhancement.

References

[1] Conway, R.J., Heinen, J.A., and Niederjohn, R.J., "Evaluation of a Technique Involving Processing with Feature Extraction to Enhance the Intelligibility of Noise-corrupted Speech," Proceedings of the 16th Annual Conference of the IEEE Industrial Electronics Society, pp. 28-33, Nov. 27-30, 1990, Pacific Grove, CA.

[2] Paliwal, K.K., "Estimation of Noise Variance from the Noisy AR Signal and Its Application in Speech Enhancement," IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-36, pp. 292-294, Feb. 1988.

[3] Cadzow, J.A., "Spectral Estimation: An Overdetermined Rational Model Equation Approach," Proc. IEEE, vol. 70, pp. 907-939, Sept. 1982.

Prefiltering and Selective Linear Prediction to Improve Formant Tracking of Noise Corrupted Speech

by Thomas J. Svoren, Russell J. Niederjohn,
James A. Heinen and Robert B. Wallace[1]
Department of Electrical and Computer Engineering,
Marquette University, Milwaukee WI 53233

Introduction

In this paper an improved method for formant tracking of voiced speech in low signal-to-noise ratios (SNR) is discussed. The basis of this new method is the use of linear prediction to independently model each of the formant frequency ranges. This is achieved by two different approaches. The first approach involves filtering the speech for each formant frequency range separately, then tracking each formant frequency individually using linear prediction (LP). The second approach involves the use of selective linear prediction (SLP) on each formant frequency range and independent determination of each formant frequency. A third technique is also considered where prefiltering is performed preceeding selective linear prediction. These approaches increase the probablity of the presence of at least one spectral peak in each of the formant frequency ranges. Improvement over several previously implemented formant tracking algorithms [1] has been obtained. These results were determined through two common error measures comparing estimated formant contours to a standard formant contour for a database of thirty-speaker-utterances. A brief review of selective linear prediction and prefiltering to achieve improved formant tracking is presented in the following sections.

Discussion

Selective linear prediction [2] can be used to model selected portions of the spectrum, rather than modeling the entire spectral range. In the application presented in this paper, the portion of the spectrum of interest is each of the formant frequency ranges of the speaker(s). These frequency ranges are known generally and are used to determine the frequency ranges for the selective linear predictor. The advantage of using SLP is two-fold. First, since a smaller range of the frequency spectrum is modeled, fewer coefficients for the predictor are needed. This results in fewer extraneous peaks. Second, SLP models only the formant frequency range being considered. This increases the probability that at least one peak will occur in each known formant frequency range.

To further enhance the formant frequency estimation algorithm, a prefilter can be used to process the speech before other processing is performed. The idea behind using this filter for formant estimation is to provide the formant tracker with speech data emphasizing the frequency range of interest. Since these filters are applied to the signal before the estimation of each formant frequency they are referred to as formant prefilters.

When designing the formant prefilters the stopband frequencies are the extremes of each formant frequency range. The gain of these filters is then set to one. Other filter shapes were considered as possible prefilters but none of these shapes resulted in any significant change in the results.

---

[1] Address: Research and Development Unit, Federal Bureau of Investigation,
Quantico, VA 22315

Once the three formant prefilters are chosen, the formant frequency extraction algorithm proceeds as follows. The original speech plus noise signal is filtered with the first formant prefilter. The resulting signal then serves as input to the formant tracking algorithm. The spectral peak having highest confidence is chosen as the first formant frequency. With the first formant determined the original signal is then filtered with the second formant prefilter from which the second formant frequency is determined in a similar manner as the first formant frequency. Finally, the third is likewise determined. By limiting the frequency range provided to the formant tracker, the linear predictor is able to more accurately model the desired frequency ranges.

Results

Two error measures were calculated on the estimated formant contours. The first is the percentage of missed formant frequencies outside a frequency band around the known formant frequency. The second is a percentage of extraneous peaks within a frequency band around the known formant frequency. The results indicate that the prefiltering approach to provides an improvement over previously implemented formant tracking algorithms. SLP alone is only slightly better than linear prediction. By prefiltering before SLP there is some improvement over using LP with prefiltering.

Conclusions

Using a frequency constrained approach on the spectrum of speech contaminated by high levels of additive noise, improvement has been obtained for formant frequency estimation over previously implemented formant tracking algorithms. The best results were achieved using selective linear prediction after prefiltering.

Acknowledgment

The authors are pleased to acknowledge the Research and Development Unit, Federal Bureau of Investigation for the partial support of this research.

References

[1] T.V.Sreenivas and R.J. Niederjohn, "Zero-Crossing Based Spectral Analysis and 'SVD' Spectral Analysis for Formant Frequency Estimation in Noise", IEEE Trans. on Signal Processing, accepted for publication.

[2] J.D. Markel and A.H. Gray Jr., Linear Prediction of Speech, New York: Springer Verlag, 1976.

[3] J. Makhoul, "Linear Prediction: A Tutorial Review", Proc. IEEE, Vol. 63, pp.561-580, 1975.

[4] N.S. Reddy and M.N.S. Swamy, "High Resolution Formant Extraction from Linear Prediction Phase Spectra", IEEE Trans. on Signal Processing , Vol. ASSP-32, No. 6, pp. 1136-1144, Dec. 1984.

Concurrent Implementation of Digital Filters

Jorge L. Aravena and Nupur Bhattacharyya
Department of Electrical and Computer Engineering
Louisiana State University, Baton Rouge, LA 70803-5901
e-mail: aravena@max.ee.lsu.edu

Abstract

This paper proposes a highly concurrent implementation for digital filters characterized by their transfer function. The method is based on an approximation of the impulse response and a representation of the filtering algorithm as a sum of small size triple matrix terms. The resulting structure has reduced arithmetic complexity and a high degree of parallelism. The use of dedicated architectures creates a fast and efficient time domain implementation. The technique is tested with FIR and IIR highpass and bandpass filters. Simulation results are presented.

1. Introduction

The problem of efficient processing has received extensive attention (see for example [1]-[8]). The representations proposed here appear to have many of the computational advantages of transform-based implementations and do not require any complex algebra manipulation.

In many applications the number of data points is finite and known beforehand. The filtering operation then is a matrix-vector multiplication. Thus, parallel implementations of linear algebraic operations can be used to develop direct *time domain implementations* of filtering algorithms with enhanced computational speed. The results in [9]-[12] describe the use of systolic arrays and a highly concurrent representation of matrix-vector products to develop fast linear algorithms. The representation, called the *triple matrix structure,* is applicable to any matrix and replaces a matrix-vector product by number of small-size matrix products that can be executed concurrently.

This paper considers the application of the triple matrix structure to the development of *time domain implementations* of IIR filtering algorithms described by discrete transfer functions. In particular, it studies the problem of *approximating* a filter with limited number of triple matrix structures -such a case arises in situations with limited amount of hardware. For the IIR filters it is possible to establish a rule to determine the number of triple matrix products required to obtain a good performance. The fixed size representation is extended to on-line operation. This last case is also used to analyze implementation and computational issues.

Section 2 gives the notation and basic results of the triple matrix representation. Section 3 states the approximation problem for the case of fixed block length. It develops a simple criterion to estimate the number of terms required for a good approximation. Experimental results comparing the exact filter and its approximations are also included. Section 4 develops extensions and presents results applicable to on-line operation. Section 5 considers implementation issues related to this technique.

2. Mathematical Preliminaries

Kronecker products are an important tool in the development of the triple matrix structure. Formally, if L and R are $p \times q$ and $k \times s$ matrices respectively then their Kronecker product, $L \otimes R$, is a $pk \times qs$ matrix with block representation

$$L \otimes R = [r_{ij} L] \, .$$

Further details and properties of the product can be found in [14]. The property which is germane to this development is described by the operation *vec* introduced below.

If Y is a $p \times q$ matrix then $y = vec(Y)$ is a pq-dimensional vector obtained by stacking the columns of Y from left to right. Consider now a pq-dimensional linear transformation $y = L \otimes R x$; $y, x \epsilon \Re^{pq}$, $L \epsilon \Re^{p \times p}, R \epsilon \Re^{q \times q}$.

Define $p \times q$ matrices, Y, X, such that $y = vec(Y)$, $x = vec(X)$. By direct computation one can show

$$Y = LXR^t \Leftrightarrow vec(Y) = [L \otimes R]vec(X);$$

where 't' denotes transpose. The computation of the vector, $y = L \otimes Rx$, with the usual row-column algorithm requires $(pq)^2$ multiplications and additions. On the other hand, the computation of $Y = LXR^t$ requires only $pq(p+q)$ operations. Furthermore, as described in [9], with specially built architectures the operation can be accomplished in $O(\sqrt{pq})$ compute cycles. For example if $p = q = 16$, the fastest proposed architecture computes the triple product in 32 cycles. By contrast, a strictly sequential implementation, of $y = L \otimes Rx$, requires 256^2 compute cycles.

For a given $pk \times qs$ matrix, T, a representation of the form

$$T = \sum_i L_i \otimes R_i \ ,$$

is called a triple matrix decomposition because $y = Tx$ can be computed using

$$Y = \sum_i L_i X R_i^t \ ; \qquad y = vec(Y), x = vec(X).$$

The decomposition always exists but it is not unique [9]. A decomposition is *minimal* if the number of its Kronecker products is the smallest among all such representations.

If the number of terms in a representation is high, a completely concurrent implementation of all matrix operations may have a prohibitive cost. In such cases the concept of approximate representation arises naturally. The quality of the approximation can be determined by its distance to the exact map. This work uses the distance induced by the matrix inner product, $< X, Y > = tr\{XY^*\}$. The superscript '$*$' indicates transpose conjugate and '$tr\{\ .\ \}$' denotes the *trace* of the enclosed matrix.

An *orthogonal canonical* triple matrix representation is a representation of the form

$$T = \sum_i \lambda_i L_i \otimes R_i \ , \qquad \lambda_i \geq \lambda_{i+1};$$

and the sets $\{ L_i \}$, $\{ R_i \}$, are each orthonormal (i. e., $< L_i, L_j > = < R_i, R_j > = \delta_i^j$).

The importance of this representation is given by the following result:

Lemma: Assume that T has the orthogonal canonical representation

$$T = \sum_{i=1}^{q} \lambda_i L_i \otimes R_i \ .$$

The approximation

$$\hat{T}_k = \sum_{i=1}^{k} \lambda_i L_i \otimes R_i \qquad (k < q)$$

is optimal in the least square sense among all triple matrix representations with k terms; i. e.,

$$\| T - \hat{T}_k \| \leq \| T - T_k \|$$

for any other T_k with k terms.

An algorithm to determine the orthogonal canonical representation is given in [10].

3. Triple Matrix Representation for IIR Filters

Consider an IIR filter described by its discrete transfer function

$$H(z) = \frac{b(z)}{a(z)} = \frac{b_0 + \sum_{j \geq 1} b_j z^j}{1 + \sum_{j \geq 1} a_j z^j} \ ;$$

the unit delay operation has transfer function $D(z) = z$. The filter is also described by

$$y_k = \sum_{j \geq 0} h_j x_{k-j} \ , \quad k = 0, 1, 2, \ldots \ .$$

Defining the vectors $\boldsymbol{y} = \mathrm{col}\ \{ y_0, y_1, \ldots \}$, $\boldsymbol{x} = \mathrm{col}\ \{x_0, x_1, ..\}$ the filter admits the form $\boldsymbol{y} = H\boldsymbol{x}$. The linear operator, H, has the infinite matrix representation

$$H = \begin{bmatrix} h_0 & 0 & 0 & . & . \\ h_1 & h_0 & 0 & . & . \\ h_2 & h_1 & h_0 & . & . \\ . & . & . & . & . \\ . & . & . & . & . \end{bmatrix} .$$

Consider first the case in which the filter is used to process a finite number, m, of data points; for the initial studies, the paper has selected $m = 256$. In a later section the paper discusses extensions for very long sequences. If the number of points is $m = n^2$, one can consider the filter described by the $n^2 \times n^2$ matrix, T, derived from the first n^2 rows and columns of H, and determine its orthogonal canonical triple matrix product decomposition. With the orthogonal canonical representation one can easily consider the optimal triple matrix approximation for T. As mentioned before, the objective is to approximate the filter with a limited number of terms in the triple product decomposition. It would be helpful if the number of terms required can be estimated from the description of the filter transfer function. Such a criterion is developed in the following.

If all the poles of $H(z)$ lie outside the disc $\| z \| = r^{-1}$ $(r < 1)$, then $| h_n | \leq c r^n$, where c is a constant[13]. Now, for the computation of the triple matrix representation, the $n^2 \times n^2$ matrix, T, is partitioned into $n \times n$ matrices as

$$T = \begin{bmatrix} T_0 & 0 & . & . & 0 \\ T_1 & T_0 & 0 & . & 0 \\ T_2 & T_1 & T_0 & . & . \\ . & . & . & . & . \\ T_{n-1} & . & . & . & T_0 \end{bmatrix} ; \ T_0 = \begin{bmatrix} h_0 & 0 & . & . & 0 \\ h_1 & h_0 & 0 & . & 0 \\ h_2 & h_1 & h_0 & . & . \\ . & . & . & . & . \\ h_{n-1} & . & . & . & h_0 \end{bmatrix} , \ T_k = \begin{bmatrix} h_{kn} & h_{kn-1} & . & . & . \\ h_{kn+1} & h_{kn} & & . & . \\ h_{kn+2} & h_{kn+1} & h_{kn} & . & . \\ . & . & . & . & h_{kn-1} \\ h_{(k+1)n-1} & . & . & h_{kn+1} & h_{kn} \end{bmatrix}$$

$$(k = 1, \ldots)$$

Since T has the lower triangular block Toeplitz form, the ratio of the norm of the blocks $\| T_k \| / \| T_0 \|$ is a good indication of the number of terms necessary for an approximate triple matrix representation of T. In fact, the number of terms in the triple product representation is bounded above by the number of non-zero blocks below the diagonal. For example, if $\| T_k \| / \| T_0 \|$ is zero for k=3,..., then T has a banded form which can be approximated with no more than 3 terms.

Given the expressions for T_0 and T_k and the bound for $| h_n |$, it is straightforward to find expressions and bounds for $\| T_0 \|$ and $\| T_k \|$ in terms of r. One can write

$$\| T_0 \|^2 = n h_0^2 + \sum_{i=1}^{n-1} (n-i) h_i^2 \ ;$$

$$\| T_0 \| \leq \| T_0(r) \|^2 = c\{n + \sum_{i=1}^{n-1} (n-i) r^{2i}\} = c \sum_{i=0}^{n-1} (n-i) r^{2i}$$

and

$$\begin{aligned} \| T_k \|^2 &= nh_{kn}^2 + \sum_{i=1}^{n-1} (n-i) \{h_{kn+i}^2 + h_{kn-i}^2\} \\ \| T_k \| \leq \| T_k(r) \|^2 &= c[nr^{2kn} + \sum_{i=1}^{n-1} (n-i) r^{2kn} \{r^{2i} + r^{-2i}\}] \\ &= cr^{2kn} \{ \sum_{i=0}^{n-1} (n-i) r^{2i} + \sum_{i=0}^{n-1} (n-i) r^{-2i} - n \} \\ &= cr^{2kn} \{ \| T_0(r) \|^2 + \| T_0(r^{-1}) \|^2 - n \}. \end{aligned}$$

Hence one can estimate the ratio as

$$\frac{\| T_k \|^2}{\| T_0 \|^2} = \frac{r^{2kn} \{ \| T_0(r) \|^2 + \| T_0(r^{-1}) \|^2 - n\}}{\| T_0(r) \|^2} .$$

It is apparent that $\| T_k \|^2 / \| T_0 \|^2$ decays as r^{2nk}. It can be concluded that stable IIR filters can be reasonably approximated with triple matrix product representations consisting of at most three terms. In fact, for $r \leq 0.9$, two terms are sufficient for a good approximation, as will be seen in the simulation results.

Remark :The above analysis does not hold for FIR filters, since there are no finite poles.

In this case, $\| T_k \| / \| T_0 \|$ is bounded above by $\lceil \frac{length\ of\ impulse\ response}{n} \rceil$.

3.1 Examples: Triple Matrix Representations of Known Filters

Using the algorithm described in [10], triple matrix product representations with one and two terms are computed for known filter transfer functions. The performance of the approximation is compared with that of the actual filter by testing with a randomly generated input signal. The filters used here are:

FIR Lowpass Filter : $H(z)=$
$0.0024 - 0.0068z^2 + 0.0145z^4 - 0.0272z^6 + 0.0494z^8 - 0.0970z^{10} + 0.3149z^{12} + 0.3149z^{14} - 0.0970z^{16} + 0.0494z^{18} - 0.0272z^{20} + 0.0145z^{22} - 0.0068z^{24} + 0.0024z^{26}$

IIR Highpass Filter :

$$H(z)=\frac{0.0005 - 0.0039z + 0.0138z^2 - 0.0276z^3 + 0.0345z^4 - 0.0276z^5 + 0.0138z^6 - 0.0039z^7 + 0.0005z^8}{1 + 3.5421z + 7.1529z^2 + 9.6066z^3 + 9.2245z^4 + 6.3924z^5 + 3.1136z^6 + 0.9781z^7 + 0.1541z^8}$$

IIR Bandpass Filter :

$$H(z)=\frac{0.0240 + 0.0024z - 0.0545z^2 - 0.0006z^3 + 0.0546z^4 - 0.0018z^5 - 0.0241z^6}{1 + 1.8612z^2 + 1.4323z^4 + 0.4077z^6} .$$

Note that z is the unit delay.

Simulation results are shown in *Figures 1 - 3* respectively. For clarity in the display and artificial bias has been introduced. The length of data sequence is taken as 256. The plots show the DFTs of the input signal, of the actual filtered signal $y=Tx$ and of the triple matrix product approximations

$$y^{(q)} = vec(Y^{(q)}) = vec(\sum_{i=1}^{q} L_i X R_i) , q= 1,2.$$

It is seen that in all the three cases, the two term approximation $y^{(2)}$ matches the actual filtered signal quite closely. This is in accordance with the analysis presented in the previous section for stable IIR filters. For the FIR filter, $h(k)=0$, $k \geq 26$. Therefore, for $n= 16$, two terms are sufficient for an exact representation of the filter.

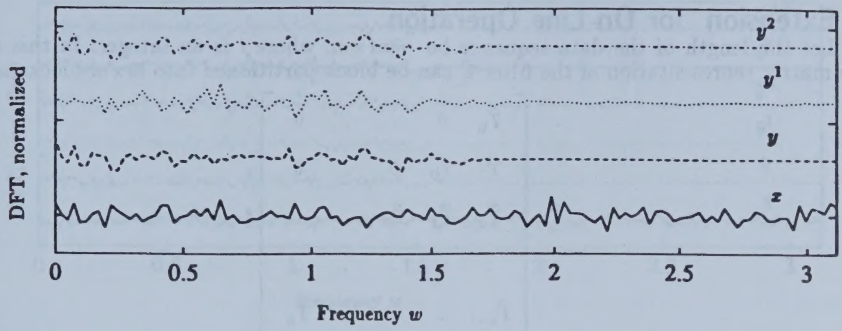


Fig. 1 FIR Lowpass Filter, m=256

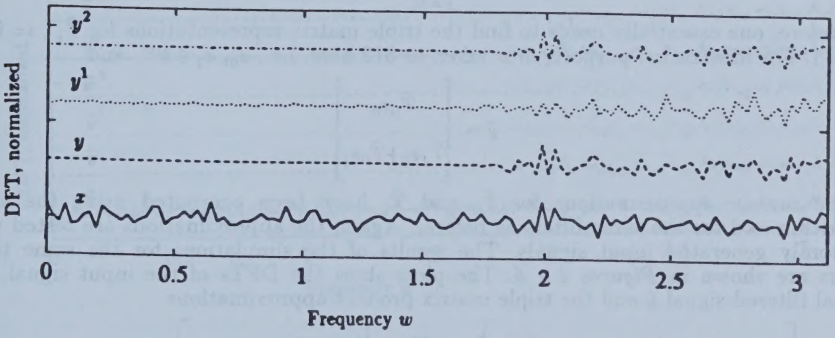


Fig. 2 IIR Highpass Filter, m=256

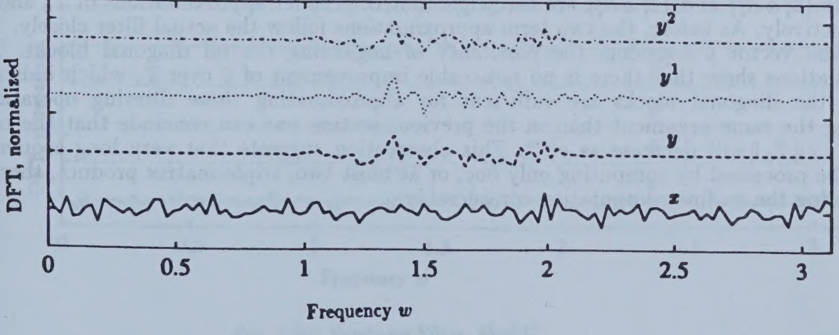


Fig. 3 IIR Bandpass Filter, m=256

4. Extension for On-Line Operation

Let the length of the data sequence be $\widehat{m}=l\times m$, where l is an integer. In this case, the matrix representation of the filter $\widehat{T}$ can be block partitioned into $m\times m$ blocks as

$$\widehat{T}=\begin{bmatrix} \widehat{T}_0 & 0 & . & . & 0 \\ \widehat{T}_1 & \widehat{T}_0 & 0 & . & 0 \\ \widehat{T}_2 & \widehat{T}_1 & \widehat{T}_0 & . & . \\ . & . & . & . & . \\ \widehat{T}_{l-1} & . & . & . & \widehat{T}_0 \end{bmatrix};$$

where $\widehat{T}_i$, $i= 0,1,\ldots,(l-1)$ are $m\times m$ block matrices. Considering the data sequences partitioned in $m-$vectors x_i , y_i , $i= 0,1,\ldots\, l-1. \ldots$, it follows that

$$\widehat{x} = col\{\, x_0, x_1, \ldots, x_{l-1}\ldots\} \quad \widehat{y} = col\{y_0, y_1, \ldots, y_{l-1}\ldots\,.\,\}$$

and

$$y_k = \sum_{j=0}^{k} \widehat{T}_{k-j} x_j \ , \quad k = 0, 1, \ldots\,.$$

Therefore, one essentially needs to find the triple matrix representations for $\widehat{T}_i$, $i= 0, 1, \ldots, l-1$. For illustration purpose, $\widehat{m}$ is taken as 512 here, i.e , $x_0, x_1 \in \Re^m$ and

$$\widehat{y} = \begin{bmatrix} \widehat{T}_0 x_0 \\ \widehat{T}_1 x_0 + \widehat{T}_0 x_1 \end{bmatrix}.$$

Triple matrix approximations for $\widehat{T}_0$ and $\widehat{T}_1$ have been computed using the same algorithm and for the same filters as before. Again, the approximations are tested with randomly generated input signals. The results of the simulations for the same three filters are shown in *Figures 4 - 6.* The plots show the DFTs of the input signal, the actual filtered signal $\widehat{y}$ and the triple matrix product approximations

$$\widehat{y}^{(q)} = \begin{bmatrix} \mathrm{vec}\{\sum_{i=1}^{q} L_i^0 X_0 R_i^0\} \\ \mathrm{vec}\{\sum_{i=1}^{q} L_i^1 X_0 R_i^1 + L_i^0 X_1 R_i^0\} \end{bmatrix}, \ \widetilde{y}^{(q)} = \begin{bmatrix} \mathrm{vec}\{\sum_{i=1}^{q} L_i^0 X_0 R_i^0\} \\ \mathrm{vec}\{\sum_{i=1}^{q} L_i^0 X_1 R_i^0\} \end{bmatrix}, \ q= 1,2.$$

where $\{L_i^0 \otimes R_i^0\}$ and $\{L_i^1 \otimes R_i^1\}$ are the triple matrix product approximations of $\widehat{T}_0$ and $\widehat{T}_1$ respectively. As before, the two term approximations follow the actual filter closely.

The vector $\widetilde{y}$ considers the possibility of neglecting the off diagonal blocks. The simulations show that there is no noticeable improvement of $\widehat{y}$ over $\widetilde{y}$, which indicates that the diagonal blocks are sufficient for approximating these filtering operations. Using the same argument than in the previous section one can conclude that the ratio $\|\widehat{T}_k\| / \|\widehat{T}_0\|$ will decrease as r^{2km}. This observation suggests that very long sequences can be processed by computing only one, or at most two, triple matrix product, thereby reducing the on line computation considerably.

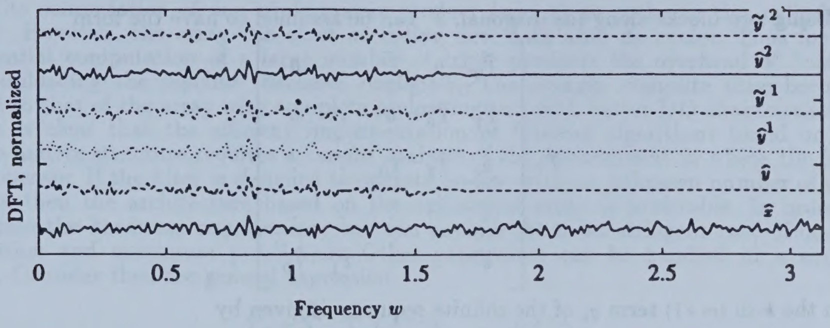


Fig. 4 FIR Lowpass Filter, $\widehat{m}$=512

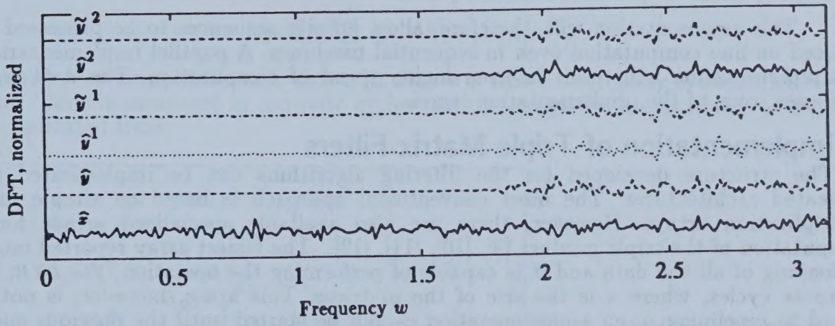


Fig. 5 IIR Highpass Filter, $\widehat{m}$=512

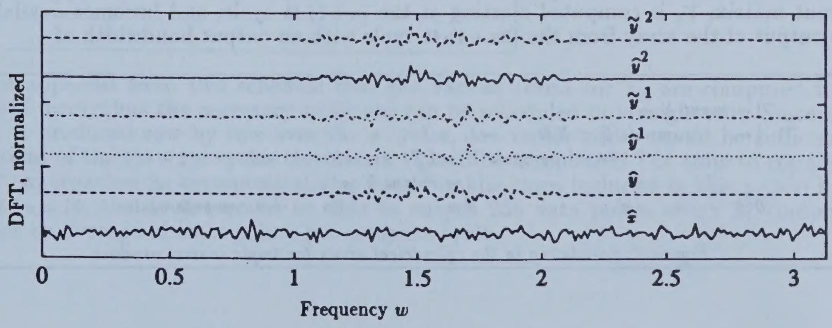


Fig. 6 IIR Bandpass Filter, $\widehat{m}$=512

Using two blocks along the diagonal, $\widehat{T}$ can be assumed to have the form

$$\widehat{T}=\begin{bmatrix} \widehat{T}_0 & 0 & . & . & 0 \\ \widehat{T}_1 & \widehat{T}_0 & 0 & . & 0 \\ 0 & \widehat{T}_1 & \widehat{T}_0 & . & . \\ . & . & . & . & . \\ 0 & . & . & \widehat{T}_1 & \widehat{T}_0 \end{bmatrix},$$

then the k-th $(m \times 1)$ term y_k of the infinite sequence is given by

$$y_k = \widehat{T}_1 x_{k-1} + \widehat{T}_0 x_k ,$$

where y_k, $x_k \in \Re^m$. The recursive expression then is approximated by

$$Y_k^{(q)} = \sum_{i=1}^{q} [L_i^1 X_{k-1} R_i^1 + L_i^0 X_k R_i^0].$$

where, based on the previous analysis, $q = 1, 2$.

This representation will, therefore, allow infinite sequences to be processed with reduced on-line computation even in sequential machines. A parallel implementation of this representation will result in even higher speed of computation. The next section addresses some of the implementation issues.

5. Implementation of Triple Matrix Filters

The structure developed for the filtering algorithms can be implemented using dedicated architectures. The most conventional approach is based on simple matrix multiplication arrays. However, there are also available specialized arrays for the computation of the triple product [9], [10], [11], [12]. The fastest array reported requires preloading of all the data and it is capable of performing the operation, $Y = LXR$, in $2n$ compute cycles, where n is the size of the matrices. This array, however, is not well suited to pipelining; i. e., a new operation cannot be started until the previous one has been completed and unloaded from the array. Furthermore, for a maximum of efficiency all data should be loaded or unloaded in parallel.

An array with better I/O requirements is based on a cylindrical architecture [9]. Details on its operation can be found in the literature. For the purpose of this paper it is enough to consider the basic scheduling as shown in *Figure 8.* The input $n \times n$ data matrix, X, is fed to the array in regular form over n cycles, with a bandwidth of n, and can be retrieved at the output (bottom) of the array with an $n-$cycles delay. The output matrix, Y, is computed starting at the $(n+1)$-th cycle, and becomes available at the output of the array from the $(2n+1)$-th cycle with an output bandwidth of n.

|← *input* X →|
|← *compute* $Z = XR$ →|
|← *compute* $Y = LZ$ →|
|← *output* Y →|
0 n *2* n $3n$ ⟶ *compute cycles*

Figure 7. Scheduling in the cylindrical array for triple matrix product

The computation of one triple matrix product requires $3n$ cycles in this cylindrical array. However, the array can start accepting new data after $2n$ cycles. Thus for the sequential computation of a large number of triple products the overhead of 'loading and unloading the pipeline' becomes negligible. The average compute time becomes equal to that of the array with complete preloading and with better I/O characteristics.

It is clear that the efficient implementation of filtering algorithms based on the triple matrix structure requires a careful analysis of the environment in which the filter will operate. If the filter is designed to operate on-line with an unknown number of data points then the architecture based on the cylindrical array is preferable. In order to estimate the hardware requirements one can assume the most stringent case of on-line operation and maximum parallelism. Other constraints can be handled in a similar form. Consider then the general expression

$$Y_k = \sum_{j=1}^{p} \sum_{i=1}^{q} L_i^{j-1} X_{k-j-1} R_i^{j-1}.$$

This expression uses p blocks of size $m \times m$ with each block approximated by q triple matrix structures of size $n \times n$; thus one would require a total of pq arrays. The examples use $p = 2$, $q = 2$; thus only four triple matrix architectures would be required.

Assuming that q arrays can receive the data in parallel one can use as basic unit the quantity

$$Y_k^{-j} = \sum_{i=1}^{q} L_i^{j-1} X_{k-j-1} R_i^{j-1}.$$

The schedule of computations is shown in *Figure 8* where each array is identified by its p value. Time is measured in compute cycles and the quantities are available at the end of the indicated time.

	$(j = 0)$	$(j = 1)$	$(j = 2)$	....
n	X_1			
$2n$		X_1		
$3n$	$X_2 \mid Y_1^0$		X_1	
$4n$		$X_2 \mid Y_2^{-1}$		
$5n$	$X_3 \mid Y_2^0$		$X_2 \mid Y_3^{-2}$	
$6n$		$X_3 \mid Y_3^{-1}$		
$7n$	$X_4 \mid Y_3^0$		$X_3 \mid Y_4^{-2}$	
.				
.				
.			$X_{k-1} \mid Y_k^{-2}$	
.		$X_k \mid Y_k^{-1}$		
$(2k+1)n$	$X_{k+1} \mid Y_k^0$			

Figure 8. Schedule of Computation for the triple matrix structure

It is apparent from this schedule that the various terms for Y_k are computed in a staggered form; thus the necessary additions can be scheduled in a single unit. Since the output is produced row by row over the n cycles, one vector adder would be sufficient. At the end of the $(2k+1)n$ cycles the matrix Y_k will be completed. The time to compute each Y_k approaches $2n$ asymptotically as $k \to \infty$. For the cases included in this paper; i.e., $m = 256$, $n = 16$, the device would be able to output 256 data points every 32 compute cycles or an average of eight points per compute cycle.

Closure

The paper has presented a new highly parallel implementation of digital filters. This implementation is based on the time domain and does not require any complex algebra manipulations. Simulation results show that with a reasonable number of triple product unit one can implement conventional filters. The number of units required can be estimated from the location of the poles of the filter transfer function.

The performance of the exact filtering algorithm and its approximations have been compared using computer simulation and fixed length blocks of 256 data points. The matrices in the triple matrix decomposition were 16×16. For the cases studied, the number of triple products required to get a good approximation is usually less than or equal to three. The recursive extensions for on-line operation was tested with block size 256 and it is capable of processing unlimited number of blocks at an average rate of eight points per compute cycle.

The method proposed permits several variations and extensions. For future reference one can consider the possibility of combining L-path filters [4] techniques. For the matrix representation of the algorithm this would imply the reordering of columns. Only the lower triangularity would be lost.

This study considered processing blocks of 256 points; thus the matrix representation of the algorithm was partitioned in blocks of that size. As another alternative one could consider reducing the number of points processed at one time. The number of hardware units would increase but the I/O requirements may be less severe. Furthermore, with units of smaller size one may be able to make more efficient implementations. For example, preliminary simulation results indicate that if the impulse response decays rapidly, one may be able to reduce the hardware requirements and still get good results using 8×8 units instead of 16×16 units.

References

[1] Ahmad , M.O. , Poornalah , D.V. , " Design of an efficient VLSI inner-product processor for real-time DSP applications " , IEEE Trans. Circ. Syst. , Vol. 36 , No.2 , Feb. 1989 , pp. 324-329.

[2] Ajmi-Sadjadi , Rostampour , " Parallel and Pipeline architectures for 2D block processing ", IEEE Trans. Circ. Syst. , Vol. 36 , No.3 , Mar. 1989 , pp. 443-448.

[3] Hussain , Jaragh , " A triangular systolic array for the discrete-time deconvolution " , IEEE Trans. Circ. Syst. , Vol. 36 , No.4 , Apr. 1989 , pp. 622-627.

[4] Acha , José , " Computational Structures for fast implementation of L-path and L-block digital filters ", IEEE Trans. Circ. Syst. , Vol. 36 , No.6 , June. 1989 , pp. 805-812.

[5] Kalaouptsidis , Theoderidis, " Fast sequential algorithms for least squares FIR filters with linear phase" , IEEE Trans. Circ. Syst. , Vol. 35 , No.4 , Apr. 1988 , pp. 425-432.

[6] Mikhael, Wu, " Fast Algorithms for block FIR Adaptive Digital Filtering ", IEEE Trans. Circ. Syst. , Vol. CAS-34 , No.10 , Oct. 1987 , pp. 1152-1160.

[7] Hayashi *et al*, " Design of high speed Digital Filter suitable for multi-DSP problems", IEEE Trans. Circ. Syst. , Vol. CAS-33 , No.2 , Feb. 1986 , pp. 202-217.

[8] Lu *et al*, " Fast recursive filtering with multiple slow processing elements ", IEEE Trans. Circ. Syst. , Vol. CAS-32 , No.11 , Nov. 1985 , pp. 1119-1129.

[9] Aravena , J.L. , " Triple Matrix Product Architectures for Fast Signal Processing", IEEE Trans. Circ. Syst. , Vol. CAS-35 , No.17 , Jan. 1988 , pp. 119-122.

[10] Aravena , J.L. , Porter , W.A. , " Array based Design of Digital Filters ", IEEE Trans. ASSP, Vol. 38 , No.9 , Sept. 1990.

[11] Porter , W.A. , Aravena , J.L." Fast 2D Signal Processing ", IEEE Trans. Proc., (Special issue on Signal Processing), Vol. 78 , No.4 , Apr. 1990 , pp. 768-778.

[12] Aravena , J.L. , Barbir , O. , " A class of Low Complexity, High Concurrence Algorithms", (to appear) IEEE Trans. Parallel. and Distrib. Syst.

[13] Hille, H., Analytic Function Theory, Ginn & Co., Boston, 1959

[14] Marcus, M., Finite Dimensional Multilinear Algebra, Part I, Marcel Dekker, Inc.,New York, 1973.

Response Classification of Floating Point Realizations of Linear Discrete Systems

P. H. Bauer and Jie Wang
Department of Electrical Engineering
Laboratory of Image and Signal Analysis
University of Notre Dame
Notre Dame, IN 46556

Abstract:

It is shown that floating point realizations of linearly stable systems can exhibit four fundamental types of free responses. Sufficient conditions for the existence or non-existence of these response types in a given system are presented.

I. Introduction

The implementation of linear discrete systems on a digital computer can be achieved using three types of arithmetic schemes:
floating point, block floating point and fixed point arithmetic. The floating point scheme is by far the most popular and is used in practically all general purpose computing devices. Its main advantage over the fixed point scheme is the high dynamic range and good robustness against quantization errors. Consequently, linear and nonlinear system simulations are mostly performed in floating point format. By choosing the mantissa and exponent register length sufficiently high, one hopes to reduce the effects of quantization to a negligible level. In most practical examples, no parasitic oscillations or other types of nonconverging responses are observed. This is due to the fact that the IEEE Floating Point Standard requires a minimum of 24 bits for the mantissa and 8 bits for the exponent. It will be shown in this paper that a linear system implemented, using the IEEE standard on floating point arithmetic, must have an extremely small margin of (linear) stability in order to show any noticable parasitic oscillations. It will be proven that granulary oscillations are likely to exist although their small amplitude will require the system to operate in an underflow regime. The questions which will be investigated focus on the existence of the various types of system responses, in particular on conditions for their existence.

A significant body of research on the effects of quantization in floating point implementations of linear systems already exists. Most of the previous work [1-9] uses a stochastic approach to the problem and provides output signal to noise ratios as a function of the linear system properties and the quantization format. Only a few pioneering papers [10,11] address the problem of a deterministic response analysis. In both approaches it is usually assumed that there is an infinite dynamic range, i.e., an infinite length exponent register which always allows for a normalization of the mantissa. This paper includes the consideration of underflow and the problems caused by it.

In Section II, we explain the notation used throughout this paper. Section III introduces bounds on the quantization error of a second order system described by a difference equation. Section IV discusses the four basic response types and provides sufficient conditions for their existence. (The occurrence of underflow will be taken into account since underflow will always happen in a real implementation if the system response is close to zero.)

Finally, Section V contains concluding remarks and ongoing research in this area.

II. Notation

Throughout this paper the following notation will be used:
A number x will be represented in floating point format as

$$\begin{aligned} x &= x^{(m)} \cdot 2^{x^{(e)}} \\ x^{(m)} &: \text{ mantissa of } x \\ x^{(e)} &: \text{ exponent of } x \end{aligned} \tag{1}$$

Except for a denormalized representation during underflow we have

$$\frac{1}{2} \leq |x^{(m)}| < 1 \tag{2}$$

Furthermore we use the notation:

l_m : mantissa register length
l_e : exponent register length
$E_{\min}$: minimum representable exponent

Floating Point Multiplication:

Let a and y be two numbers represented in floating point format as described above. Then

$$Flp(a \cdot y) = a^{(m)} \cdot y^{(m)} \cdot 2^{a^{(e)}+y^{(e)}} \tag{3}$$

The result is stored using a single length product mantissa. Consequently the mantissa product has to be quantized. It will be assumed that quantization is performed using a truncation scheme. (Two's Complement or Magnitude Truncation.)

Floating Point Addition:

Let p_1 and p_2 be two numbers represented in floating point format as described above. Then the floating point addition $Flp(p_1 + p_2)$ is carried out in 3 steps:

1. Mantissa denormalization of the smallest term (if necessary) and exponent adjustment. Without loss of generality, assume $p_2^{(e)} \geq p_1^{(e)}$:

$$\begin{aligned} p_2 &= p_2^{(m)} \cdot 2^{p_2^{(e)}} \\ \tilde{p}_1 &= \underbrace{p_1^{(m)} \cdot 2^{p_1^{(e)}-p_2^{(e)}}}_{\tilde{p}_1^{(m)}} \cdot 2^{p_2^{(e)}} \end{aligned} \tag{4}$$

$\tilde{p}_1^{(m)}$ denotes the denormalized mantissa of $\tilde{p}_1$.

2. Addition:

$$p = \tilde{p}_1 + p_2 = \left(\tilde{p}_1^{(m)} + p_2^{(m)}\right) \cdot 2^{p_2^{(e)}} = p^{(m)} \cdot 2^{p^{(e)}} \tag{5}$$

3. Sum mantissa normalization (if necessary):

$$p_s = \underbrace{(p^{(m)} \cdot 2^{-1})}_{p_s^{(m)}} \cdot 2^{p^{(e)}+1} = p_s^{(m)} \cdot 2^{p_s^{(e)}} = Flp(p_1 + p_2) \tag{6}$$

Consequently three types of quantization errors have to be considered:

ϵ_{PQ} : product quantization error

ϵ_{MD} : reformating error due to mantissa denormalization and exponent adjustment

ϵ_{MN} : reformating error due to possible mantissa normalization after summation

For simplicity reformating is performed by truncation in all cases. (Again, two's complement or magnitude truncation will be considered.) This includes the most simple option of just disregarding the least significant bits.

Before quantifying various reformating error bounds, we will explain what is considered underflow within the framework of this paper. Since we consider a linear system implemented in the time-domain, all recursions consist of the computation of a sum of products, each product consisting of two factors. The system operates in an underflow regime, if underflow occurs in *each* product. Underflow in a product occurs if either one of the factors requires a denormalized mantissa representation or if the product requires a denormalized mantissa representation.

III. Error Bounds

In this section error bounds for $\epsilon_{PQ}, \epsilon_{MN}$ and ϵ_{MD} are derived under the assumption that no underflow occurs. Then upper bounds on the same errors are derived for underflow conditions. Using the Bibo property of the ideal linear system, these bounds on the reformating error are used to compute a bound on the system response.

Next we will describe reformating errors:

Product Quantization:

$$|\epsilon_{PQ}| = |Flp(a \cdot y) - a \cdot y| < 2^{-l_m} \cdot 2^{a^{(e)}+y^{(e)}} \tag{7}$$

Denormalization:

$$|\epsilon_{MD}| < 2^{-l_m + p_2^{(e)}}, \tag{8}$$

where $p_2^{(e)} = \max\{p_1^{(e)}, p_2^{(e)}\}$ in Eq. (4).

Normalization:

$$|\epsilon_{MN}| < 2^{-l_m-1} \cdot 2^{p_2^{(e)}+1} = 2^{-l_m+p_2^{(e)}} \tag{9}$$

(A maximum of a single digit right shift is possible for the normalization of the mantissa of a two term sum.)

The above reformating error bounds are valid, if underflow is avoided in each of the three reformating schemes.

We will now consider error bounds valid for underflow:

Product Quantization:

$$|\epsilon_{PQ}| < 2^{-l_m} \cdot 2^{E_{\min}} \text{ if } a^{(e)} \leq 0 \qquad \text{(product underflow)}$$

$$|\epsilon_{PQ}| < 2^{-l_m} \cdot 2^{E_{\min}+a^{(e)}} \text{ if } a^{(e)} > 0 \text{ and } y^{(e)} = E_{\min} \qquad \text{(signal underflow)}$$

$$\text{Therefore: } |\epsilon_{PQ}| < 2^{-l_m} \cdot 2^{\max(a^{(e)}+E_{\min}, E_{\min})} \tag{10}$$

Mantissa Denormalization:

$$|\epsilon_{MD}| < 2^{-l_m+p_2^{(e)}}, \quad p_2^{(e)} \geq E_{\min} \tag{11}$$

Mantissa Normalization:

$$|\epsilon_{MN}| < 2^{-l_m+p_2^{(e)}}, \quad p_2^{(e)} \geq E_{\min} \tag{12}$$

Throughout this paper we will consider the following zero input second order nonlinear difference equation to represent the system under consideration:

$$y(n) = Flp(Flp(a_1 y(n-1)) + Flp(a_2 y(n-2))) \tag{13}$$

(A generalization to higher order systems is feasible but it is omitted at this point to keep the notation to a minimum.)

It will be further assumed that the coefficients a_1, a_2 are within the socalled stability triangle in order to ensure stability of the linear model. Although we will treat the case of underflow due to its importance for zero convergence, we will not allow overflow to occur. In other words, it is assumed that the dynamic range of the system is sufficiently high relative to the system response.

Using the previously established notation (see Section II) Eq. (13) is said to operate in an underflow regime if underflow conditions happen in both products $Flp(a_1 y(n-1))$ and $Flp(a_2 y(n-2))$. Underflow conditions are said to happen in a product, if either the signal $y(n-1)$ or $y(n-2)$ requires a denormalized mantissa or the product requires a denormalized mantissa, which means that either the signal or the product exponent has to be $E_{\min}$. (From the definition in Section II, the possibility of a denormalized coefficient mantissa also exists. Since this is highly unlikely to occur, we do not allow this case, i.e., coefficients smaller than $0.5 \cdot 2^{E_{\min}}$.)

With the previously established error bounds, Eq. (13) can be rewritten as a linear equation driven by an error term, which incorporates all the reformating errors:

$$y(n) = a_1 y(n-1) + a_2 y(n-2) + \delta(n-1) \tag{14}$$

where $\delta(n)$ is the input signal generated by the various reformating errors.

Next we will present bounds on $|\delta(n)|$ using the previously established individual error bounds. Since in some cases, a reformating error absorbs the one from a previous operation, the bound on $|\delta(n)|$ is actually lower than the sum of the individual bounds.

For the following error formulation we will use the notation:

$$y_B^{(e)} \geq y^{(e)}(n), \forall n \tag{15}$$

$y_B^{(e)}$ is therefore a bound on the exponent of the signal $y(n)$ for all times n. We also define:

$$a_{\max} = \max(|a_1|, |a_2|). \tag{16}$$

A simple analysis and (7-9) yield:

$$|\delta(n)| < 2^{-l_m+a_{\max}^{(e)}+y_B^{(e)}} + 2^{-l_m+a_{\max}^{(e)}+y_B^{(e)}} + 2^{-l_m+a_{\max}^{(e)}+y_B^{(e)}} = 3 \cdot 2^{-l_m+a_{\max}^{(e)}+y_B^{(e)}} \tag{17}$$

In (17) the first term represents the product quantization error of the largest possible product. The second term stands for the product quantization error and the denormalization error of the smaller product. Observe that this error bound is equal to the denormalization error, since for truncation the denormalization error absorbs the previously occuring product quantization error.

The third term describes the error which might occur due to normalization of the mantissa of the sum. Equation (17) is valid if the system in (13) does not operate in an underflow regime.

For underflow conditions, we obtain from (10-12):

$$|\delta(n)| < 3 \cdot 2^{-l_m} 2^{\max\{E_{\min}, E_{\min}+a^{(e)}_{\max}\}} \tag{18}$$

As in the case without underflow, the denormalization error absorbs the quantization error of the smallest term, since we always assume the necessity of denormalization for a worst-case analysis.

In the following section, the boundedness of the input (error) signal in (17) and (18) will be used to obtain bounds on the system response.

IV. The Basic Response Types

This section investigates the types of responses which can be generated from a linear system implemented in floating point format. Since equation (14) is an equivalent formulation of (13), we will use the linear form (14) for our analysis.

Unlike in the fixed point case, the bound on $|\delta(n)|$ is not independent of the system outputs. From (17) we have

$$|\delta(n)| < R \cdot 2^{y_B^{(e)}} \tag{19}$$

where R is some real constant, depending on the floating point and quantization format. Therefore, the error bound increases with the magnitude of the system output $y(n)$. If a system with an infinite dynamic range is assumed (no underflow, no overflow), an unbounded response can be generated if the feedback gain, relating $y(n)$ and $\delta(n)$ is sufficiently high. If the feedback gain over the whole dynamic range is sufficiently small, asymptotic convergence will occur. Looking at equations (14) and (19) as an output feedback problem, we have

$$\delta(n) = f(y(n)) = k \cdot y(n), \tag{20}$$

where k is an uncertain gain. There are plenty of tools available to solve this problem.

Unfortunately, the real situation is somewhat more complicated:

The main reason is the occurence of underflow. In an underflow regime, the error bound for $|\delta(n)|$ cannot be reduced, if the signal's amplitude reduces. Therefore, equation (20) is not valaid for underflow. Furthermore, reformating errors caused by truncating the least significant bits do not tend to zero if $y(n)$ tends to zero. The popular approximations of floating point operations

$$Flp(a+b) = (a+b)(1+\delta_1) \tag{21}$$

and

$$Flp(a \cdot b) = a \cdot b(1+\delta_2) \tag{22}$$

do therefore not hold for underflow conditions since in underflow $|\delta_2| << 1$ cannot be satisfied. (Most of the results based on the above approximations depend on δ_1 and δ_2 to be significantly smaller than 1.) In fact, in the presence of underflow δ_2 could, for example, be as large as $\delta_2 = -1$, causing the floating point product to vanish.

Clearly with the simplifying assumption of an infinite dynamic range, i.e., an exponent register of infinite length, one expects three types of zero-input responses:

- Type 1: unbounded, nonperiodic response, caused by an increasing error bound on $\delta(n)$.
- Type 2: a bounded response.
- Type 3: an asymptotically stable response, i.e., a response which converges to zero.

If a system with a limited dynamic range is considered, one can distinguish between two types of nonzero, bounded responses, one of which operates in an underflow regime:

- R1-type: An increasing system response, bounded only by the finite system dynamics (overflow).
- R2-type: A bounded, periodic response, (which does not cause overflow).
- R3-type: A granulary, periodic system response, where the system operates in an underflow regime.
- R4-type: An asymptotically stable response, where the output reaches zero after a finite number of recursions.

Figure 1 illustrates the relationship between the four possible response classes.

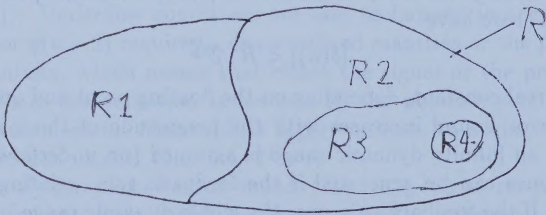


Figure 1: Response classes of floating point linear systems.

For a given linear system, the quantization format, the mantissa length and the initial conditions will determine in which class a particular response will fall. Observe that the set of responses R4 is a subset of R3, which again is a subset of R2. The sets R1 and R2 partition the total set R of all possible responses.

Next we will provide conditions which will guarantee response types R2 and R3. These sufficient conditions will guarantee response types R2 and R3 independent of the choice of initial conditions.

Theorem 1:

The free response of the floating point system realization

$$Flp(Flp(a_1 y(n-1) + Flp(a_2 y(n-2))) = y(n) \tag{23}$$

belongs to the class R2, if the mantissa register length satisfies:

$$l_m \geq a_{\max}^{(e)} + \log_2 3 + \log_2 H + 1,$$

where $H = \sum_{n=0}^{\infty} |h(n)|$, with $h(n)$ being the impulse response of the linear system (14).

Theorem 2:

The free response of the floating point system realization

$$Flp(Flp(a_1 y(n-1)) + Flp(a_2 y(n-2))) = y(n) \tag{24}$$

belongs to the class R3 and is asymptotically bounded in magnitude by

$$|y(n)|_{n\to\infty} < 2^{\max(E_{\min}, E_{\min} - a_{\max}^{(e)}) - 2}$$

if the mantissa register length satisfies:

$$l_m \geq a_{\max}^{(e)} + \log_2 3 + \log_2 H + 2$$

with H as defined in Theorem 1.

Proof of Theorems 1 and 2:

From Eq. (14) we get

$$|y(n)| = |\delta(n) * h(n)| < \sum_{k=0}^{\infty} h(k) \cdot \sup_{0 \leq k \leq n-1} \delta(k)$$

Let y_{B_0} be the bound of the output of Eq. (13) for $0 \leq k \leq n-1$, we therefore can write:

$$|\delta(k)| < 2^{-l_m + a_{\max}^{(e)} + \log_2 3 + y_{B_0}^{(e)}} \text{ for } 0 \leq k \leq n-1.$$

Therefore

$$|y(n)| < 2^{-l_m + a_{\max}^{(e)} + \log_2 3 + y_{B_0}^{(e)}} \sum_{k=0}^{+\infty} |h(k)| = 2^{-l_m + a_{\max}^{(e)} + \log_2 3 + y_{B_0}^{(e)} + \log_2 H}$$

where $H = \sum_{k=0}^{+\infty} |h(k)|$.

Let y_{B_1} denote the bound on $|y(n+N_1)|$, N_1 sufficiently large, which is given by,

$$y_{B_1} = 2^{-l_m + a_{\max}^{(e)} + \log_2 3 + \log_2 H + y_{B_0}^{(e)}}.$$

Denote y_{B_1} as $y_{B_1}^{(m)} \cdot y_{B_1}^{(e)}$ where $y_{B_1}^{(m)}$ and $y_{B_1}^{(e)}$ are mantissa and exponent of y_{B_1}, respectively.

Now we obtain:

$$y_{B_1}^{(m)} \cdot 2^{y_{B_1}^{(e)}} = 2^{-l_m + a_{\max}^{(e)} + \log_2 3 + \log_2 H + y_{B_0}^{(e)}}$$

i.e.,

$$\begin{aligned} \log_2 y_{B_1}^{(m)} + y_{B_1}^{(e)} &= -l_m + a_{\max}^{(e)} + \log_2 3 + \log_2 H + y_{B_0}^{(e)} \\ \Longleftrightarrow y_{B_1}^{(e)} - y_{B_0}^{(e)} &= -l_m + a_{\max}^{(e)} + \log_2 3 + \log_2 H - \log_2 y_{B_1}^{(m)} \end{aligned}$$

In order to prove Theorem 2 we procede as follows: If

$$y_{B_1}^{(e)} - y_{B_0}^{(e)} \leq -1$$

we can guarantee that the response of system (13) belongs to type R3, since the same argument can be applied again: $y_{B_i}^{(e)} - y_{B_{i-1}}^{(e)} \leq -1$, where the y_{B_i} are bounds for sufficiently large time instances $N_i, N_i >> N_{i-1}$.

Hence we have

$$l_m \geq a_{\max}^{(e)} + \log_2 3 + \log_2 H - \log_2 y_{B_1}^{(m)} + 1 \tag{25}$$

Since $\frac{1}{2} \leq y_{B_1}^{(m)} < 1$ we get

$$-1 \leq \log_2 y_{B_1}^{(m)} < 0$$

Therefore, if

$$l_m \geq a_{\max}^{(e)} + \log_2 3 + \log_2 H + 2 \tag{26}$$

the magnitude of response of system (13) will decrease gradually and can be asymptotically bounded by

$$|y(n)|_{n \to \infty} < 2^{\max(E_{\min}, E_{\min} - a_{\max}^{(e)}) - 2}$$

which is obtained from the bound in (18).

Comment:

The proof of Theorem 1 can easily be found by changing the condition $y_{B_1}^{(e)} - y_{B_0}^{(e)} \leq -1$ to $y_{B_1}^{(e)} - y_{B_0}^{(e)} \leq 0$. □

For a given reformating scheme, the mantissa length is dependent only on the l_1 measure of the impulse response of the linear system and the maximum coefficient. The margin of stability of the linear system enters the condition through H. The two conditions in Theorem 1 and 2 show, that if the condition for an R2 response is satisfied, then by just adding one bit in the mantissa register one can ensure a R3 type response, which is usually sufficient. Also observe, that the response bound in Theorem 2 implies, that either the response or the product terms in the difference equation reach the minimum exponent E_{min}. Therefore the system operates on the boundary of or in an underflow regime.

Considering that the largest possible coefficient exponent of a stable second order linear difference equation is equal to 1 and that the minimum mantissa length according to the IEEE Floating Point Format is equal to 24, the condition

$$log_2 H \leq 20$$

ensures a R3 type response. The above condition is practically always satisfied, even in systems with very little margin of linear stability.

V. Conclusion

The four basic response types for floating point implementations of linear systems under zero input were introduced. Sufficient conditions for the existence of two response types were formulated and a bound for R3 class responses was derived. It is demonstrated that zero input limit cycles are practically not observable in second order linear systems, if one follows the IEEE Floating Point Format. All possible limit cycles will create underflow conditions.

Although we concentrated our efforts on a second order difference equation, the same principle applies to any linear system. Conditions for the existence of R1 to R4 type responses in general linear systems are currently under investigation.

References

[1] B. Liu and T. Kaneko, "Error Analysis of Digital Filters Realized with Floating Point Arithmetic", *Proc. IEEE*, Vol. 57, pp. 1735-1747, Oct. 1969.

[2] C. Weinstein and A.V. Oppenheim, "A Comparison of Roundoff Noise in Floating Point and Fixed Point Digital Filter Realizations", *Proc. IEEE*, Vol. 57, pp. 1181-1183, June 1969.

[3] M. Akamine and T. Higuchi, "State Space Approach to Synthesis of Minimum Quantization Digital Filters Using Floating Point Arithmetic", *IEEE Symposium on Circuits and Systems*, pp. 1002-1005, 1984.

[4] T. Kaneko and B. Liu, "On Local Roundoff Error in Floating Point Arithmetic", *J. ACM*, Vol. 20, No. 3, pp. 391-398, July 1973.

[5] B. Liu, "Effect of Finite Wordlength on the Accuracy of Digital Filters - A Review", *IEEE Trans. on Circuit Theory*, Vol. CT-18, No.6, pp. 670-677, Nov. 1971.

[6] J. Barlow, "On the Distribution of Accumulated Roundoff Errors in Floating Point Arithmetic", *Proc. 5th Sym. Computer Arithmetic*, 18-19 May 1981, Ann Arbor, MI, pp. 100-105.

[7] L.M. Smith, B.W. Bomar, R.D. Joseph, G.C. Yang, "Floating Point Roundoff Noise Analysis of Second Order State Space Digital Filters", submitted to *IEEE Trans. on Circuits and Systems.*

[8] B. Zeng and Y. Neuvo, "Roundoff Noise Analysis for Floating Point Digital FiltersRealized in Cascade and Parallel Forms", *Proc. Int. Conf. on Circuits and Systems*, pp. 898-901, Nanjjing 1989.

[9] B. Zeng and Y. Neuvo, "Analysis of Floating Point Roundoff Errors Using Dummy Multiplier Coefficient Sensitivity", to appear in *IEEE Trans. on Circuits and Systems.*

[10] I. W. Sandberg, "Floating Point Roundoff Accumulation in Digital Filter Realizations", *Bell Systems Tech. J.*, Vol. 46, pp.1775-1791, Oct. 1967.

[11] T. Kaneko, "Limit-Cycle Oscillations in Floating Point Digital Filters",*IEEE Trans. on Audio and Electroacoustics*, Vol. AU-21, No.2, April 1973, pp.100-106.

The Influence of Quantization Formats on Absolute Error Bounds in Floating Point Autoregressive Models

P. H. Bauer and Jie Wang
Laboratory of Image and Signal Analysis
Department of Electrical Engineering
University of Notre Dame
Notre Dame, IN 46556

Abstract:

This paper addresses the effect of various floating point arithmetic reformating schemes on the convergence of autoregressive difference equations. Truncation and rounding quantization schemes as well as double and single length product mantissa schemes will be analyzed and compared.

I. Introduction

In [1], convergence and divergence criteria were derived for second order recursive difference equations implemented in floating point format. In order to reduce the number of cases to be considered, it was assumed that the floating point format has a single length product mantissa and that all reformating operations (product quantization, normalization, denormalization) applied a simple bit truncation scheme. Obviously many other possibilities exist. This paper is concerned with conditions for output convergence to zero (R4) or to oscillations, with extremely small amplitudes (R3). It was established in [1], that only two of the four possible response types (R1-R4) of floating point systems are acceptable responses (R3 and R4).

In this paper, we answer the following two questions:
Are their any reformating operations, which minimize the number of required mantissa bits in order to ensure a R3 or R4 type response? By how much do the mantissa register length requirements differ for various formats, i.e., are significant savings in the mantissa register length possible by choosing an optimum format?

This paper uses the same notation as [1]. For a more detailed discussion on quantization and reformating error bounds the reader is also referred to [1]. Next, we give a brief outline of this article:

In section II, we briefly present the notation and formulate error bounds pertinent to the considered floating point formats. In section III, the main results are derived, interpreted and discussed. Finally, section IV contains concluding remarks and directions for further research.

II. Preliminaries

Throughout this paper the following notation will be used:
A number x will be represented in floating point format as

$$\begin{aligned} x &= x^{(m)} \cdot 2^{x^{(e)}} \\ x^{(m)} &: \text{ mantissa of } x \\ x^{(e)} &: \text{ exponent of } x \end{aligned} \qquad (1)$$

Except for a denormalized representation during underflow we have

$$\frac{1}{2} \leq |x^{(m)}| < 1 \tag{2}$$

Furthermore we use the notation:

l_m : mantissa register length

$E_{\min}$: minimum representable exponent

Floating Point Multiplication:

Let a and y be two numbers represented in floating point format as described above. Then

$$Flp(a \cdot y) = a^{(m)} \cdot y^{(m)} \cdot 2^{a^{(e)}+y^{(e)}} \tag{3}$$

The result is stored using a single length or double length product mantissa. In the single length case, the mantissa product has to be quantized. It will be assumed that quantization is performed using either a truncation scheme (Two's complement, Magnitude Truncation) or a roundoff scheme.

Floating Point Addition:

Let p_1 and p_2 be two numbers represented in floating point format as described above. Then the floating point addition $Flp(p_1 + p_2)$ is carried out in 3 steps:

1. Mantissa denormalization of the smallest term (if necessary) and exponent adjustment. Without loss of generality, assume $p_2^{(e)} \geq p_1^{(e)}$:

$$\begin{aligned} p_2 &= p_2^{(m)} \cdot 2^{p_2^{(e)}} \\ \tilde{p}_1 &= \underbrace{p_1^{(m)} \cdot 2^{p_1^{(e)}-p_2^{(e)}}}_{\tilde{p}_1^{(m)}} \cdot 2^{p_2^{(e)}} \end{aligned} \tag{4}$$

   $\tilde{p}_1^{(m)}$ denotes the denormalized mantissa of $\tilde{p}_1$.

2. Addition:

$$p = \tilde{p}_1 + p_2 = \left(\tilde{p}_1^{(m)} + p_2^{(m)}\right) \cdot 2^{p_2^{(e)}} = p^{(m)} \cdot 2^{p^{(e)}} \tag{5}$$

3. Sum mantissa normalization (if necessary):

$$p_s = \underbrace{(p^{(m)} \cdot 2^{-1})}_{p_s^{(m)}} \cdot 2^{p^{(e)}+1} = p_s^{(m)} \cdot 2^{p_s^{(e)}} = Flp(p_1 + p_2) \tag{6}$$

4. Sum quantization: This step is necessary only in the double length case.

Consequently three types of quantization errors have to be considered:

ϵ_{PQ} : product quantization error (single length case only)

ϵ_{MD} : reformating error due to mantissa denormalizatoin and exponent adjustment

ϵ_{MN} : reformating error due to possible mantissa normalization after summation

ϵ_{SQ} : sum quantization (double length case only)

Before quantifying various reformating error bounds, we will explain what is considered underflow within the framework of this paper. Since we consider a linear system implemented in the time-domain, all recursions consist of the computation of a sum of products, each product consisting of two factors. The system operates in an underflow regime, if underflow occurs in *each* product. Underflow in a product occurs if either one of the factors requires a denormalized mantissa representation or if the product requires a denormalized mantissa representation.

Next error bounds for ϵ_{PQ}, ϵ_{MN}, ϵ_{MD} and ϵ_{SQ} are formulated under the assumption that no underflow occurs. Then upper bounds on the same errors are derived for underflow conditions. Using the Bibo property of the ideal linear system these bounds on the reformating error are later used to compute a bound on the system response. We will describe reformating errors as following:

Product Quantization (Not applicable for double length format)

- Single length, truncation scheme:

$$|\epsilon_{PQ}^{T1}| < 2^{-l_m} \cdot 2^{p_2^{(e)}} \tag{7}$$

where $p_2^{(e)} = \max\{p_1^{(e)}, p_2^{(e)}\}$ in Eq. (4).

- Single length, rounding scheme:

$$|\epsilon_{PQ}^{R1}| \leq 2^{-l_m-1} \cdot 2^{p_2^{(e)}} \tag{8}$$

Denormalization

- Single length, truncation scheme:

$$|\epsilon_{MD}^{T1}| < 2^{-l_m+p_2^{(e)}}, \tag{9}$$

- Double length, truncation scheme:

$$|\epsilon_{MD}^{T2}| < 2^{-2l_m+p_2^{(e)}} \tag{10}$$

- Single length, rounding scheme:

$$\epsilon_{MD}^{R1}| \leq 2^{-l_m-1+p_2^{(e)}} \tag{11}$$

- Double length, rounding scheme:

$$|\epsilon_{MD}^{R2}| \leq 2^{-2l_m-1+p_2^{(e)}} \tag{12}$$

Normalization

- Single length, truncation scheme:

$$|\epsilon_{MN}^{T1}| \leq 2^{-l_m-1} \cdot 2^{p_2^{(e)}+1} = 2^{-l_m+p_2^{(e)}} \tag{13}$$

- Double length, truncation scheme:

$$|\epsilon_{MN}^{T2}| \leq 2^{-2l_m-1} \cdot 2^{p_2^{(e)}+1} = 2^{-2l_m+p_2^{(e)}} \tag{14}$$

- Single length, rounding scheme:

$$|\epsilon_{MN}^{R1}| \leq 2^{-l_m-1} \cdot 2^{p_2^{(e)}+1} = 2^{-l_m+p_2^{(e)}} \tag{15}$$

- Double length, rounding scheme:

$$|\epsilon_{MN}^{R2}| \leq 2^{-2l_m-1} \cdot 2^{p_2^{(e)}+1} = 2^{-2l_m+p_2^{(e)}} \tag{16}$$

Sum Quantization (Not applicable for single length format)

- Double length, truncation scheme:

$$|\epsilon_{SQ}^{T2}| \leq 2^{-l_m+p_2^{(e)}+1} \tag{17}$$

- Double length, rounding scheme:

$$|\epsilon_{SQ}^{R2}| \leq 2^{-l_m-1} \cdot 2^{p_2(e)+1} = 2^{-l_m+p_2(e)} \tag{18}$$

The above reformating error bounds are valid, if underflow is avoided in each of the three reformating schemes.

We will now consider error bounds valid for underflow:

Product Quantization in Underflow (Not applicable for a double length format)

- Single length, truncation scheme:

$$|\epsilon_{PQ}^{T1}| < 2^{-l_m} \cdot 2^{\max(a^{(e)}+E_{\min},E_{\min})} \tag{19}$$

- Single length, rounding scheme:

$$|\epsilon_{PQ}^{R1}| \leq 2^{-l_m-1} \cdot 2^{\max(a^{(e)}+E_{\min},E_{\min})} \tag{20}$$

Denormalization in Underflow

- Single length, truncation scheme:

$$|\epsilon_{MD}^{T1}| < 2^{-l_m+p_2^{(e)}}, \;\; p_2^{(e)} \geq E_{\min} \tag{21}$$

- Double length, truncation scheme:

$$|\epsilon_{MD}^{T2}| < 2^{-2l_m+p_2^{(e)}} \tag{22}$$

- Single length, rounding scheme:

$$|\epsilon_{MD}^{R1}| \leq 2^{-l_m-1+p_2^{(e)}} \tag{23}$$

- Double length, rounding scheme:

$$|\epsilon_{MD}^{R2}| \leq 2^{-2l_m-1+p_2^{(e)}} \tag{24}$$

Normalization in Underflow

- Single length, truncation scheme:

$$|\epsilon_{MN}^{T1}| \leq 2^{-l_m+p_2^{(e)}} \tag{25}$$

- Double length, truncation scheme:

$$|\epsilon_{MN}^{T2}| \leq 2^{-2l_m+p_2^{(e)}} \tag{26}$$

- Single length, rounding scheme:

$$|\epsilon_{MN}^{R1}| \leq 2^{-l_m+p_2^{(e)}} \tag{27}$$

- Double length, rounding scheme:

$$|\epsilon_{MN}^{R2}| \leq 2^{-2l_m+p_2^{(e)}} \tag{28}$$

Sum Quantization in Underflow (Not applicable for single length format)

- Double length, truncation scheme:

$$|\epsilon_{SQ}^{T2}| < 2^{-l_m+p_2^{(e)}+1} \tag{29}$$

- Double length, rounding scheme:

$$|\epsilon_{SQ}^{R2}| \leq 2^{-l_m-1+p_2^{(e)}+1} = 2^{-l_m+p_2^{(e)}} \tag{30}$$

The following second order recursive model will be considered:

$$y(n) = a_1 y(n-1) + a_2 y(n-2) \tag{31}$$

If Eq. (29) is implemented in floating point arithmetic, the following formats are of interest:

Format 1:

$$y(n) = Flp_T(Flp_T(a_1 y(n-1)) + Flp_T(a_2 y(n-2))) \tag{32}$$

where Flp_T stands for floating point multiplication or addition, using truncation as the reformating scheme. Therefore in the above equation the products are quantized (to single length mantissas) using a truncation scheme. Normalization and denormalization also use truncation.

Format 2:

$$y(n) = Flp_T(Flp_R(a_1 y(n-1)) + Flp_R(a_2 y(n-2))) \tag{33}$$

where Flp_R stands for floating point multiplication or addition, using rounding as the reformating scheme. Therefore, after multiplication the mantissa products are rounded to a single length format. The other operation are the same as in Format 1.

Format 3:

$$y(n) = Flp_R(Flp_R(a_1 y(n-1)) + Flp_R(a_2 y(n-2))) \tag{34}$$

This is the same as format 1, except that rounding is performed in each operation.

Format 1-3 all used a single length mantissa format, since mantissa quantization occurred right after multiplication. The following schemes are double length mantissa formats, i.e., there is no quantization after the multiplication operation. Instead quantization occurs after (or together with) normalization of the sum:

Format 4:

$$y(n) = Flp_T(a_1 y(n-1) + a_2 y(n-2)) \tag{35}$$

This is the same as format 1, except for the multiplications being performed in full precision, i.e., using double length mantissa registers. The final sum is quantized to a single length mantissa using truncation.

Format 5:

$$y(n) = Flp_R(a_1y(n-1) + a_2y(n-2)) \tag{36}$$

This format produces the smallest total quantization error, since all reformating operations are done using rounding and no product quantization occurs.

It will be further assumed that the coefficients a_1, a_2 are within the so-called stability triangle in order to ensure stability of the linear model. Although we will treat the case of underflow due to its importance for zero convergence, we will not allow overflow to occur. In other words, it is assumed that the dynamic range of the system is sufficiently high relative to the system response.

Using the previously established notation (see Section II) eq. (31) is said to operate in an underflow regime if underflow conditions happen in both products $a_1y(n-1)$ and $a_2y(n-2)$. Underflow conditions are said to occur in a product, if either the signal $y(n-1)$ or $y(n-2)$ requires a denormalized mantissa or the product requires a denormalized mantissa, which means that either the signal or the product exponent has to be $E_{\min}$. (From the definition in Section II, the possibility of a denormalized coefficient mantissa also exists. Since this is highly unlikely to occur, we do not allow this case, i.e., coefficients smaller than $0.5 \cdot 2^{E_{\min}}$.)

With the previously established error bounds, the five recursive models can be rewritten as a linear equation driven by an error term, which incorporates all the reformating errors:

$$y(n) = a_1y(n-1) + a_2y(n-2) + \delta(n-1) \tag{37}$$

where $\delta(n)$ is the input signal generated by the various reformating errors.

Next we will present bounds on $|\delta(n)|$ using the previously established individual error bounds. Since in some cases, a reformating error absorbs the error from a previous operation, the bound on $|\delta(n)|$ is in some cases lower than the sum of the individual bounds.

For the following error formulation we will use the notation:

$$y_B^{(e)} \geq y^{(e)}(n), \forall n \tag{38}$$

$y_B^{(e)}$ is therefore a bound on the exponent of the signal $y(n)$ for all times n. We also define:

$$H = \sum_{n=0}^{+\infty} |h(n)| \tag{39}$$

$$a_{\max} = \max(|a_1|, |a_2|) \tag{40}$$

We will now formulate the total reformating error in formats 1-5. For the single length product mantissa scheme, this total reformating error is the sum of the product quantization error, mantissa denormalization error and the mantissa normalization error. In the case of a double length registers, the total error is the sum of the mantissa denormalization error, the mantissa normalization error and the sum quantization error. Note the error created by the first operation is sometimes absorbed by the error of the second operation. With this in mind, we arrive at the following total reformating error bounds for formats 1-5.

Format 1:

From [1] it is known that

$$|\delta(n)| < 3 \cdot 2^{-l_m + a^{(e)}_{\max} + y^{(e)}_B} \tag{41}$$

and for underflow conditions we obtain:

$$|\delta(n)| < 3 \cdot 2^{-l_m} 2^{\max\{E_{\min}, E_{\min} + a^{(e)}_{\max}\}} \tag{42}$$

Format 2:

From Eq. (8), (9) and (13) we obtain:

$$|\delta(n)| = |\epsilon^{R1}_{PQ}| + |\epsilon^{T1}_{MD}| + |\epsilon^{T1}_{MN}| < (2^{-l_m-1} + 2^{-l_m} + 2^{-l_m}) 2^{a^{(e)}_{\max} + y_B(e)} = 2.5 \cdot 2^{-l_m + a^{(e)}_{\max} + y^{(e)}_B} \tag{43}$$

and for underflow conditions this becomes:

$$|\delta(n)| < 2.5 \cdot 2^{-l_m} 2^{\max\{E_{\min}, E_{\min} + a^{(e)}_{\max}\}} \tag{44}$$

Format 3:

From Eq. (8), (11) and (15) we obtain:

$$|\delta(n)| = |\epsilon^{R1}_{PQ}| + |\epsilon^{R1}_{MD}| + |\epsilon^{R1}_{MN}| \leq (2^{-l_m-1} + 2^{-l_m-1} + 2^{-l_m}) 2^{a^{(e)}_{\max} + y^{(e)}_B} = 2^{-l_m + 1 + a^{(e)}_{\max} + y^{(e)}_B} \tag{45}$$

and for underflow conditions this becomes:

$$|\delta(n)| \leq 2^{-l_m+1} 2^{\max\{E_{\min}, E_{\min} + a^{(e)}_{\max}\}} \tag{46}$$

Format 4: From Eq. (10), (14) and (17) we obtain:

$$|\delta(n)| = |\epsilon^{T2}_{MD}| + |\epsilon^{T2}_{SQ}| < 2^{-2l_m} 2^{a^{(e)}_{\max} + y^{(e)}_B} + 2^{-l_m} 2^{a^{(e)}_{\max} + y^{(e)}_B + 1} = (1 + 2^{-l_m-1}) \cdot 2^{-l_m + 1 + a^{(e)}_{\max} + y^{(e)}_B} \tag{47}$$

and for underflow conditions this becomes:

$$|\delta(n)| < (1 + 2^{-l_m-1}) \cdot 2^{-l_m+1} 2^{\max\{E_{\min}, E_{\min} + a^{(e)}_{\max}\}} \tag{48}$$

Format 5: From Eq. (12), (16) and (18) we get:

$$|\delta(n)| = |\epsilon^{R2}_{MD}| + |\epsilon^{R2}_{SQ}| \leq 2^{-2l_m-1} 2^{a^{(e)}_{\max} + y^{(e)}_B} + 2^{-l_m} 2^{a_{\max} + y^{(e)}_B} = (1 + 0.5 \cdot 2^{-l_m}) \cdot 2^{-l_m + a^{(e)}_{\max} + y^{(e)}_B} \tag{49}$$

and for underflow conditions this becomes:

$$|\delta(n)| \leq (1 + 0.5 \cdot 2^{-l_m}) \cdot 2^{-l_m} 2^{\max\{E_{\min}, E_{\min} + a^{(e)}_{\max}\}} \tag{50}$$

III. Main Results

In this section, conditions on the mantissa length and asymptotic response bounds will be derived. These condition will be analyzed for the five formats discussed previously. The asymptotic response bounds force the system to operate in an underflow regime in all five cases. Since the proofs are all based on the same idea as the proofs of Theorems 1 and 2 in [1], only brief sketches are provided in this paper. For the sake of a simple comparison between the different formats, we restate the following result:

Theorem 1 [1]:

The free response of the second order autoregressive model

$$y(n) = Flp_T(Flp_T(a_1y(n-1)) + Flp_T(a_2y(n-2)))$$

is asymptotically bounded in magnitude by

$$|y(n)|_{n\to\infty} \leq 2^{\max(E_{\min}, E_{\min}-a^{(e)}_{\max})-2}$$

if the mantissa register length satisfies

$$l_m \geq a^{(e)}_{\max} + log_2 3 + log_2 H + 2$$

This result corresponds to the recursive system implemented in Format 1, which has the largest reformating error bound.

In the following Theorem, the corresponding condition is presented for a format 2 implementation:

Theorem 2:

The free response of the second order autoregressive model

$$y(n) = Flp_T(Flp_R(a_1y(n-1)) + Flp_R(a_2y(n-2)))$$

is asymptotically bounded in magnitude by

$$|y(n)|_{n\to\infty} \leq 2^{\max(E_{\min}, E_{\min}-a^{(e)}_{\max})-2}$$

if the mantissa register length satisfies

$$l_m \geq a^{(e)}_{\max} + log_2 2.5 + log_2 H + 2$$

Proof:

The proof is similar to the one of Theorem 2 in [1]. Using the bound on $|\delta(n)|$ in Eq. (43) and the consecutive bound reduction procedure in [1], one can obtains:

$$l_m \geq a^{(e)}_{\max} + log_2 2.5 + log_2 H + 2$$

From the bound on $|\delta(n)|$ for underflow conditions (44), the same asymptotic response bound as in the case of format 1 is derived.

Comment:

Comparing format 2 with format 1, it is obvious that the reduction in the number of mantissa bits is $log_2 3 - log_2 2.5 \approx 0.263$. Since l_m has to be an integer, this can in some cases lead to an improvement by a maximum of 1 bit.

Next we consider the case of rounding in each reformating operation, i.e., format 3.

Theorem 3:

The free response of the second order autoregressive model

$$y(n) = Flp_R(Flp_R(a_1y(n-1)) + Flp_R(a_2y(n-2)))$$

is asymptotically bounded in magnitude by

$$|y(n)|_{n\to\infty} \leq 2^{\max(E_{\min}, E_{\min}-a^{(e)}_{\max})-2}$$

if the mantissa register length satisfies

$$l_m \geq a^{(e)}_{\max} + \log_2 H + 3$$

Proof:

The proof can be obtained from the one of Theorem 2 in [1], by using Eq. (45) to obtain the condition on the mantissa length and Eq. (46) to produce the response bound.

Comment:

Comparing with the previously considered single length register schemes, format 3 has the smallest bound. The difference between format 1 and format 3 is $log_2 3 - 1 \approx 0.58$ bits.

We will now investigate two formats, which use a double length product mantissa. They are expected to perform better than the corresponding single length schemes. At first format 4 will be considered.

Theorem 4:

The free response of the second order autoregressive model

$$y(n) = Flp_T(a_1 y(n-1) + a_2 y(n-2))$$

is asymptotically bounded in magnitude by

$$|y(n)|_{n\to\infty} \leq 2^{\max(E_{\min}, E_{\min} - a^{(e)}_{\max}) - 2}$$

if the mantissa register length satisfies:

$$l_m \geq a^{(e)}_{\max} + log_2 H + 3$$

Proof:

The somewhat conservative total error bound is given by equation (47). This yields the mantissa length condition of the theorem. The bound on the response is obtained as in Theorem 1-3.

Comment:

Formats 3 and 4 yield the same mantissa length conditions. This can be interpreted as follows:

In format 4, the product quantization error is zero and therefore smaller than in format 3. However, the reformating errors which follow are larger than in format 3, which uses rounding. As a result, the total error bound comes out to be constant.

Theorem [5] addresses the optimal case. Rounding is used at each stage of the process and no product quantization error can occur due to a double length format.

Theorem 5:

The free response of the second order autoregressive model

$$y(n) = Flp_R(a_1 y(n-1) + a_2 y(n-2))$$

is asymptotically bounded in magnitude by

$$|y(n)|_{n\to\infty} \leq 2^{\max(E_{\min}, E_{\min} - a^{(e)}_{\max}) - 2}$$

if the mantissa register length satisfies

$$l_m \geq a_{\max}^{(e)} + log_2 H + 2 + log_2(1 + 0.5 \cdot 2^{-l_m})$$

Proof:
Same as in Theorem 4 by using Eq. (48) and (49).

Comment:
Unfortunately, the condition in Theorem 5 is of implicit form. It is easily observed that $l_m \geq a_{\max}^{(e)} + log_2 H + 2 + log_2(1 + 0.5 \cdot 2^{-l_m}) > a_{\max}^{(e)} + log_2 H + 2$ and that the contribution of $log_2(1 + 0.5 \cdot 2^{-l_m})$ is usually very small. Consequently, if one compares formats 1 and 5, the mantissa length requirement differs maximally by $log_2 3 \approx 1.58$. In other words, the maximum savings obtainable by using different reformating schemes is 2 bits. In the following section, simulation results will be presented, which illustrate the conditions and bounds derived in this section.

IV. Conclusion

For various quantization, normalization and denormalization schemes in floating point implementations of second order difference equations, conditions on the mantissa register length have been derived, which ensure a proper zero input response behavior of the system. The scheme producing the largest reformating error and the scheme with the smallest reformating error were analyzed and the resulting mantissa length requirements were compared.

The result showed, that even for the best reformating schemes, the savings on the number of mantissa register bits is small:
One can save a maximum of 2 bits. Considering the amount of additional hardware which is required to implement the scheme with the smallest total reformating error, it appears that simply increasing the mantissa register length by 2 bits is by far preferable.

It, however, remains to be seen, whether the same conclusion hold for the general n^{th} order case. More work on this topic is warranted since floating point processors become increasingly popular in the implementation of signal processing tasks. Future work should address the general n^{th} order case for both the free and forced response case. For second order systems, the forced response case was considered in [2].

References

[1] P. H. Bauer and J. Wang, "Response Classification of Floating Point Realizations of Linear Discrete Systems", *Proceedings of the 29th Allerton Conference*, Oct. 1991.

[2] I. W. Sandberg, "Floating Point Roundoff Accumulation in Digital Filter Realizations", *Bell Systems Tech. J.* , Vol. 46, pp.1775-1791, Oct. 1967.

Adaptive Trellis and DPCM Coding of Composite Processes

Mort Naraghi-Pour Manju Hegde Sujan Das Namit Arora
Dept. Of Electrical and Computer Engineering
Louisiana State University, Baton Rouge, LA 70803

Abstract

Adaptive trellis coding and differential pulse code modulation (DPCM) of regenerative composite sources is investigated. The state of the switch in the composite source is estimated using a MAP sequence estimation algorithm and the code is then matched to the mode process corresponding to the estimated switch state. This method is used for both trellis coding and DPCM encoding of composite sources. In addition, a second method for trellis coding is presented in which the encoder implicitly estimates the switch state during the encoding process and adapts the code accordingly. Simulation results show that adaptive schemes often significantly outperform the optimized nonadaptive schemes.

I. Introduction

Composite random processes are often suggested as good models for physical sources such as speech and image data which exhibit a *multimodal* behavior. In these physical sources the local characteristics (or mode) appear to change from time to time in a random fashion. Composite sources are capable of capturing this property and in this respect they offer a distinct advantage over autoregressive processes.

A regenerative composite source [1] consists of a collection of *mode generators*, a *switch process* and a *switch mechanism*. The switch process $\{S_k\}$ is a stationary ergodic random process with alphabet $A_S = \{1,2,...,M\}$. The mode generators are devices that can be turned on and off. When turned on, the i^{th} mode generator randomly produces symbols according to some probability distribution (stationary or not) on $\mathbb{R}^{\infty}$ (the set of all doubly infinite sequences $(...,x_{-1},x_0,x_1,...)$ with $x_i \in \mathbb{R}$ for all i) which is called the i^{th} mode distribution, and which specifies the distribution of a hypothetical random process called the i^{th} mode process. More specifically, the operation of a regenerative composite source can be described as follows : when a switch changes to position j from position i, the switch mechanism turns off the i^{th} mode generator, turns on the j^{th} and directs the latter's symbols to the output. Turning on the j^{th} mode generator causes it to produce symbols according to the j^{th} distribution for as long as the switch is in position j. Although successive symbols from the j^{th} mode generator may be correlated, they are independent of all events that occurred prior to the time the j^{th} mode generator was turned on, including those involving its own previous outputs. Thus if the switch changes state at time k, all subsequent composite source outputs are conditionally independent of all past outputs. The name regenerative derives from this [1].

In this paper we assume that the switch process is a first-order Markov process with alphabet $A_S = \{1,2,...,M\}$ and irreducible state transition matrix $W = [w(i\,|\,j)]$, $i,j=1,2,...,M$. The mode processes are stationary first-order autoregressive processes, i.e.,

$$U_{n+1}^{(i)} = \rho^{(i)}\, U_n^{(i)} + W_{n+1}^{(i)} \quad , \quad i=1,2,...,M$$

where $\{W_n^{(i)}\}$, $i = 1,2,...,M$ are independent, identically distributed (iid) Gaussian random processes with mean zero and variance one and $\rho^{(i)}$'s are correlation coefficients, $|\rho^{(i)}| < 1$, $i = 1,2,...,M$. Each $U_n^{(i)}$ is Gaussian with mean zero and variance $\sigma^2_{U^{(i)}} = 1/[1-(\rho^{(i)})^2]$ and $E[\,U_n^{(i)}\, U_{n+1}^{(i)}\,] = \rho^{(i)}\sigma^2_{U^{(i)}}$. The conditional density function of $U_{n+1}^{(i)}$

given $U_n^{(i)}$ is given by

$$p^{(i)}(\alpha \mid \beta) = \frac{1}{\sqrt{2\pi}} \exp\left[\frac{-(\alpha-\rho^{(i)}\beta)^2}{2}\right].$$

We assume that the mode processes are independent of each other.

To illustrate and compare the performance of the schemes presented in this paper a composite source with two mode processes is selected. The mode processes are first-order autoregressive processes with correlation coefficients $\rho^{(1)}$ and $\rho^{(2)}$. The switch process is a symmetric two state Markov chain with $w(1|2) = w(2|1) = \delta$. All the schemes presented in this paper are tested on this source. Simulation results presented in subsequent sections are obtained using a sequence of at least 100,000 samples from the source.

II. Switch state estimation

The main idea in our approach is the following. Due to the changes in the switch state, the short term statistics of the composite source $\{X_k\}$ change. If $S_k = i$ for $n \le k \le m$, then $X_k = U_k^{(i)}$ for $n \le k \le m$, and thus for the time instants $k = n, n+1, \cdots, m$, the process $\{X_k\}$ exhibits the same statistical characteristics as the i^{th} mode process $\{U_k^{(i)}\}$. Therefore, if we can estimate the present state of the switch process, we can then adapt a given data compression scheme to the mode process (corresponding to the estimated switch state) currently being observed at the output of the composite process. The observable part of a composite process is its output process $\{X_k\}$. The estimation of the switch state should therefore be based on the past observations of $\{X_k\}$. We employ a maximum likelihood sequence estimation rule (Viterbi algorithm) for this task. This is described in the following.

At time i, the encoder receives x_i (where x_i is the realization of X_i) and is also aware of $\{\ldots x_{i-2}, x_{i-1}\}$. The maximum a posteriori (MAP) decision rule would choose that s_1^N which maximizes $p(s_1^N \mid s_0, x_1^N)$, where s_0 is the initial switch state. We can write

$$p(s_1^N \mid s_0, x_1^N) = \frac{p(x_1^N \mid s_0, s_1^N)\, p(s_1^N \mid s_0)}{p(x_1^N \mid s_0)}.$$

Since the input sequence into the encoder is the same for all switch sequences s_1^N, it is sufficient for the switch estimator to choose that s_1^N for which $p(x_1^N \mid s_0, s_1^N)\, p(s_1^N \mid s_0)$ is maximum. Utilizing the Markovness of the switch process and the mode processes we can write this as

$$p(x_1^N \mid s_0, s_1^N)\, p(s_1^N \mid s_0) = \prod_{j=1}^{N} p(x_j \mid x_{j-1}, s_{j-1}, s_j)\, w(s_j \mid s_{j-1}), \tag{1}$$

where using the regenerative property of the source we can write

$$p(x_j \mid x_{j-1}, s_{j-1}, s_j) = \begin{cases} p^{(s_j)}(x_j \mid x_{j-1}) & \text{for} \quad s_j = s_{j-1} \\ p^{(s_j)}(x_j) & \text{for} \quad s_j \ne s_{j-1}, \end{cases}$$

and where by convention $p(x_1 | x_0, s_0, s_1) = p^{(s_1)}(x_1)$. Alternatively, we can maximize

$$\sum_{j=1}^{N} \left[\, \log w(s_j \mid s_{j-1}) + \log p(x_j \mid x_{j-1}, s_{j-1}, s_j)\, \right]. \tag{2}$$

It can be seen that this may be implemented using a maximum likelihood sequence estimation rule, i.e., the Viterbi algorithm [2]. For this we construct the trellis corresponding to the state diagram of the switch process. We then employ the Viterbi algorithm for the estimation of the states of the switch process using this trellis. The metric on the branches connecting states s_{j-1} and s_j is given by

$$\log w(s_j \mid s_{j-1}) + \log p(x_j \mid x_{j-1}, s_{j-1}, s_j) . \tag{3}$$

The knowledge of the estimated switch state is then passed on to the encoder which then adapts the code accordingly.

III. Trellis source codes

For a given source, the design of a trellis code involves the selection of a trellis structure, the selection of the encoder and the design of the decoder. The trellis structure is produced by a finite state machine which, in this paper, is specialized to be a shift register containing the k most recent channel symbols. In this case the trellis structure has 2^k states. The content of the shift register is used (as a table index) to select the decoder output reproduction.

The design of a decoder involves the selection of branch labels or reproduction symbols. Many procedures have been suggested in the literature for the design of the decoder in a trellis source code (see [3] and the references therein). The procedure used here is the Linde-Buzo-Gray (LBG) algorithm tailored to trellis source code design and described in detail in [3].

For a given decoder, the encoder is completely specified by a trellis search algorithm. The search algorithm has a copy of the reproduction symbols, and attempts to find the sequence of channel symbols which minimizes the average distortion between the given source sequence and the sequence of branch labels corresponding to that channel sequence. In this paper we choose the Viterbi algorithm as the trellis search algorithm and the squared error as the distortion measure.

Trellis coding of composite sources

Two methods are considered.

Method 1.

In this method we first fix the code rate R and the length k of the shift register (2^k states in the trellis). For the resulting trellis structure we then obtain the optimal set of reproduction symbols (branch labels) for each mode process (equivalently for each switch state) using the LBG algorithm. The encoder and decoder now operate as follows. Both the encoder and decoder utilize the same trellis structure. At each instant i the encoder receives the source symbol x_i as well as the estimated switch state $\hat{s}_i$ obtained from the procedure described in Section II. The encoder then utilizes the set of branch labels corresponding to $\hat{s}_i$ (or the corresponding mode process) to encode x_i. In addition to the channel symbols, the encoder also transmits the transitions in the switch state. From these transitions the decoder reconstructs the sequence of estimated switch states and utilizes the appropriate set of branch labels to reconstruct the source sequence from the sequence of channel symbols.

Method 2.

In this scheme as in Method 1 we design one set of branch labels for each mode process. The decoder for the composite source will also operate as in Method 1. The encoder however, does not obtain the switch state estimation from the algorithm in Section II. Instead, this information is obtained during the encoding process. For this a different trellis structure is utilized by the encoder. The encoder trellis has M component trellises each with its own set of branch labels (corresponding to each mode process). New branches are added to connect the states in one component trellis to those in other component trellises. This is achieved as follows. If state α in a component trellis is connected to states $\beta_1, \beta_2, \ldots, \beta_{2^R}$ in that trellis, we then connect state α to the same set of states $\beta_1, \beta_2, \ldots, \beta_{2^R}$ in every other component trellis through additional branches. The branch labels selected for these additional branches are the same as the branch labels of the component trellis on which these branches end. This structure is similar to a rate $R \log_2 M$ trellis structure. The encoder now uses the Viterbi algorithm with this trellis structure for the encoding of the composite source. The idea is that for as long as the switch process remains in one state, the corresponding component trellis (with its own set of branch labels) is used by the encoder. However, when a transition occurs in the switch state, the component trellis used by the encoder is also changed accordingly. The difference between this and Method 1 is that in this method the squared error between the source symbol and the reproduction symbols is used as a criterion for switch estimation (i.e., to change between different component trellises) as opposed to the estimation rule of Section II.

In addition to the R channel bits per source symbol, the encoder also transmits side information about the transitions between the component trellises. This side information is utilized by the decoder to select the appropriate set of branch labels.

Figures 1 (for $\delta = .1$) and 2 (for $\delta = .01$) show the rate distortion performance of the codes of Methods 1 and 2 as well as the performance of a fixed (non-adaptive) trellis code optimally designed for the composite source. Also included in these figures are the performance of a code which has the exact knowledge of the switch state. In order to make the comparison fair, we have added an overhead rate equal to the entropy of the switch process to the rate of the code of Methods 1 and 2 as well as the code which has the exact knowledge of the switch states. It can be seen that the code of method 2 consistently outperforms the optimized nonadaptive code. The performance of Method 1 however, is somewhat disappointing. We believe this is due to the fact that although after a switch transition the branch labels are adapted to the current mode process, the state of the code is not the appropriate state and this contributes to the degradation of the code's performance.

IV. DPCM codes

Let $\{X_n\}$ be a stationary source. A DPCM code with a first-order predictor with prediction coefficient a and quantizer Q produces $\hat{X}_n$ as the reproduction for X_n according to the rule,

$$\hat{X}_n = a\hat{X}_{n-1} + Q(X_n - a\hat{X}_{n-1}) \,. \tag{4}$$

Here $\hat{X}_{n-1}$ is the code state at time n and $a\hat{X}_{n-1}$ is the prediction for X_n. The prediction error process $\{Z_n\}$ is given by $Z_n = X_n - a\hat{X}_{n-1}$ and the reproduction error process $\{E_n\}$ is given by $E_n = X_n - \hat{X}_n = Z_n - Q(Z_n)$.

The output of the quantizer is mapped into a binary word (of length $\lceil \log L \rceil$ if the quantizer has L output levels, where $\lceil x \rceil$ is the smallest integer no smaller than x), using a binary encoder, and transmitted to the decoder. We assume that this transmission is error free. The decoder, upon the receipt of this binary word will first produce the output of the quantizer using the inverse of the binary encoder. It then uses (4) to produce, recursively, the sequence $\{\hat{X}_n\}$ as the reproduction for $\{X_n\}$.

In the following by a matched DPCM code for a given first-order autoregressive source we mean a code whose predictor is matched to the source (the prediction coefficient is equal to the correlation coefficient [4,5])and whose quantizer is the Max [6] quantizer for the prediction error.

Optimized DPCM for composite sources

The optimal design of a DPCM code for a given input process involves the selection of the prediction coefficient and the quantizer so as to minimize distortion. Several attempts have been made for the optimal design of DPCM for autoregressive sources [4,5,6,7]. However, the simultaneous optimal design of the predictor and the quantizer has not been carried out.

It is generally believed that for a given predictor, the optimal quantizer is the Max quantizer for the prediction error process. Therefore, one may consider a design method where a value is arbitrarily chosen for the prediction coefficient a, the optimal quantizer is then found for this prediction coefficient and the resulting distortion registered. The value of a is then changed and the process repeated. The "optimal" value of a, denoted a_{opt}, is finally found by comparing the resulting distortions. Since the choice of the quantizer affects the distribution of the prediction error, to optimize the quantizer for a given prediction coefficient, an iterative method is often used whereby a quantizer is initially selected, for this quantizer the distribution of the prediction error is obtained, the quantizer is then optimized for this new distribution and the process repeated until the quantizer no longer changes.

The above method will be computationally complex if all values of a between zero and one are to be tested. However, a good guess for the optimal value of a can be obtained from the following argument. It is reasonable to assume that a good choice for a is that which minimizes the variance of the input into the quantizer, namely the variance of the prediction error Z_n. This variance can be written as

$$\begin{aligned} \mathrm{var}(Z_n) &= E[X_n - a\hat{X}_{n-1}]^2 \\ &= \sum_{i,j=1}^{M} E[(X_n - a\hat{X}_{n-1})^2 \mid S_{n-1}=i,\, S_n=j]\, P(S_{n-1}=i,\, S_n=j) \\ &= \sum_{i,j=1}^{M} E[(U_n^{(j)} - a\hat{U}_{n-1}^{(i)})^2 \mid S_{n-1}=i,\, S_n=j]\, P(S_{n-1}=i,\, S_n=j) \\ &= \sum_{i,j=1}^{M} E[U_n^{(j)} - a\hat{U}_{n-1}^{(i)}]^2\, \pi(i)\, w(j\mid i)\ , \end{aligned}$$

where $\pi(i)$, $i = 1,2,\ldots,M$, denotes the stationary distribution of the switch process. In the case of small distortion, $\hat{U}_n^{(i)}$ is close in distribution to $U_n^{(i)}$. Therefore, we can write

$$\mathrm{var}(Z_n) \simeq \sum_{i,j=1}^{M} \pi(i)\, w(j\mid i)\, E[U_n^{(j)} - a\, U_{n-1}^{(i)}]^2 . \tag{5}$$

Now minimizing the right-hand side of the above with respect to a we obtain a value a^* given by

$$a^* = \frac{\sum_{i=1}^{M} \pi(i)\, w(i\mid i)\, E[U_{n-1}^{(i)}\, U_n^{(i)}]}{\sum_{i=1}^{M} \pi(i)\, E[U_{n-1}^{(i)}]^2} . \tag{6}$$

Replacing the expected values for the mode processes we get a^*. (Note that for the special case that the composite source has only one mode with correlation coefficient $\rho^{(1)}$, we get $a^* = \rho^{(1)}$, which is the case of a matched predictor.) The value of a^* obtained from (6) is then used as an initial value for the prediction coefficient in the recursive method described before to obtain the optimal code.

Figure 3 shows the code distortion as a function of the prediction coefficient a for a composite source with $\rho^{(1)} = .2$, $\rho^{(2)} = .95$ and $\delta = .1$ and $.01$. It should be noted that for each value of

a, the quantizer (which has $L = 8$ output levels) is the Max quantizer for the prediction error. The values of a^* and a_{opt} are also given.

It is worth pointing out that, as demonstrated in Figure 3, the performance of the code is fairly robust to small variations of the prediction coefficient in all cases we have examined. (Also see [5].)

Adaptive DPCM for composite sources

An adaptive scheme is proposed in [8] whereby the prediction coefficient and the quantizer are changed to match the mode process which is currently present at the output of the composite source. For this the switch state is estimated from the algorithm of Section II. The knowledge of the estimated switch state is then passed on to the encoder which then chooses the DPCM code (the quantizer and the predictor) matched to the mode process corresponding to the estimated switch state.

This scheme is used for encoding of the composite source with correlation coefficients $\rho^{(1)} = .2$ and $\rho^{(2)} = .95$ [8]. Unfortunately, the results which are shown in Table 1 are somewhat disappointing. In the case of $\delta = .1$, the optimized DPCM outperformed the adaptive scheme for all values of $L = 4,8$ and 16. Only for small values of δ such as $\delta = .01$ and $\delta = .001$ and $L = 4,8$ did the adaptive scheme slightly outperform the optimized DPCM. The situation was even worse for the composite source with coefficients $\rho^{(1)} = .2$ and $\rho^{(1)} = .99$. In the following we explain the reason for this poor performance of adaptive scheme and describe a modification of the code which alleviates this problem.

Suppose the input process into a DPCM code is a first-order autoregressive process. When $L \geq 4$ and the predictor is matched to the correlation coefficient of the input process, the prediction error is very close to a zero-mean unit-variance Gaussian random variable irrespective of the value of the correlation coefficient [7]. As a result, the two quantizers used in the adaptive scheme are nearly the same as the Max quantizer for a unit-variance Gaussian random variable. Now suppose that at time n a switch transition occurs from the first mode process to the second, i.e., $S_{n-1} = 1$ and $S_n = 2$, and that this switch transition is detected by our estimator. The source outputs at times $n-1$ and n are given by $X_{n-1} = U_{n-1}^{(1)}$ and $X_n = U_n^{(2)}$. The code state at time n is given by $\hat{X}_{n-1} = \hat{U}_{n-1}^{(1)}$ and the prediction for X_n is given by $a\,\hat{U}_{n-1}^{(1)}$ where $a = \rho^{(2)}$. Now it follows from the regenerative property of the source that X_n is independent of X_i for $i < n$. Therefore, X_n is also independent of $\hat{X}_i$ for $i < n$, since the latter is a function of X_i for $i < n$. In particular, $X_n = U_n^{(2)}$ is independent of the prediction $a\,\hat{U}_{n-1}{}^{(1)}$, and therefore, the latter does not provide a good prediction for the former. In fact, the prediction error $Z_n = U_n^{(2)} - a\,\hat{U}_{n-1}^{(1)}$ is now close in distribution to a Gaussian random variable with variance $\sigma_{Z_n}^2 \approx \mathrm{var}(U_n^{(2)}) + a^2\,\mathrm{var}(U_{n-1}^{(1)})$ (assuming that $\hat{U}_{n-1}^{(1)}$ is "close" in distribution to $U_{n-1}^{(1)}$). However, as mentioned previously, the quantizer is nearly the same as a Max quantizer for a unit variance Gaussian random variable. As a result, the quantization error associated with this sample, X_n, is very large. As an example, consider the case of a composite source with $\rho^{(1)} = .2$ and $\rho^{(2)} = .95$ which gives $\sigma_{Z_n}^2 \approx 11.063$. Quantizing this random variable Z_n (with a quantizer which is very close to a Max quantizer for a unit-variance Gaussian random variable) obviously results in a large distortion. Assuming that Z_n is a zero-mean Gaussian random variable with variance 11.06 and that the quantizer is an 8-level Max quantizer for a unit-variance Gaussian random variable, we have calculated the resulting average distortion for X_n to be 3.51. We have also obtained this distortion through simulation and the result is 3.49. In this example if the switch transition is from mode 1 to mode 2, then the corresponding distortions for the first sample following the switch transition from computation and simulation are .059 and .063, respectively. The problem is obviously more pronounced for the subsource with larger variance.

The effect of the bad prediction is not limited to the first sample after a switch transition, but it lingers on for several samples. Figures 4-a and 4-b show the average distortion associated with the k^{th} source sample following a switch transition as a function of k for each of the mode processes. The larger than average distortion associated with samples following a switch transition contributes a great deal to the overall distortion of the code and degrades the code's performance particularly when the switch transitions are more frequent (larger δ). This phenomenon was also observed in [9]. To circumvent this problem we propose the following modifications. Suppose that a switch transition occurs at time n, i.e., $S_{n-1} \neq S_n$. Now since given that $S_{n-1} \neq S_n$, X_n is conditionally independent of X_i and $\hat{X}_i$ for $i < n$, the mean square error prediction of X_n based on $\hat{X}_i$ for $i < n$ is its mean value $E[X_n] = 0$. Therefore, our first modification is to improve the prediction of X_n by setting the code state after each switch transition to zero. With this modification the prediction error at time n is given by $Z_n = X_n$. Now the optimum quantizer for this prediction error is the Max quantizer for X_n which is a zero-mean Gaussian random variable whose variance is the variance of the mode which is currently on. This predictor and quantizer are used for only one sample following each switch transition. After this first sample we use the code that is matched to the current mode process. With this modification, the transient distortions following a switch transition are significantly reduced. This results in a significant improvement in the overall average distortion of the code. The transient distortions after the modification are depicted in Figures 5-a and 5-b. Table 1 also shows the distortion for this modified scheme. It can be seen that with this modification the adaptive scheme significantly outperforms the nonadaptive scheme.

References

[1] Carter, M.J., "Source Coding of Composite Sources," Ph.D. Dissertation, The University of Michigan, 1984.

[2] Viterbi A.J. and J.K. Omura, *Principles of Digital Communication and Coding,* McGraw-Hill Co., New York, NY, 1979.

[3] Viterbi A.J. and J.K. Omura, *Principles of Digital Communication and Coding,* McGraw-Hill Co., New York, NY, 1979. Stewart, R.M. Gray and Y. Linde, "The design of waveform coders," IEEE Trans. on Commun., vol. COM-30 No. 4, pp. 702-710, April 1982.

[4] Arnstein, D.S., "Quantization Error in Predictive Coders," *IEEE Trans. on Commun.,* Vol. COM-23, pp. 423-429, April 1975.

[5] M. Naraghi-Pour, and D. Neuhoff, "Mismatched DPCM Encoding of Autoregressive Processes," *IEEE Trans. on Inform. Theory,* vol. 36, no. 2, pp. 296-304, March 1990.

[6] Max, J., "Quantizing for Minimum Distortion," *IRE Trans. on Inform. Theory,* Vol. IT-6, pp. 7-12, 1960.

[7] Janardhanan, E., "Differential PCM Systems," *IEEE Trans. on Communications,* Vol. COM-27, pp. 82-93.

[8] N. Arora, M. Naraghi-Pour and M. Hegde, "Adaptive DPCM Encoding of Regenerative Composite Processes," Proceedings of the Twenty-fifth Annual Conference on Information Sciences and Systems, .pp 24-28, The Johns Hopkins University, March 20-22, 1991.

[9] Carter, M.J. and D.L. Neuhoff, "DPCM Encoding of Regenerative Composite Sources," *Proceedings, Nineteen Annual Allerton Conference on Commun., Control and Computing,* 1981.

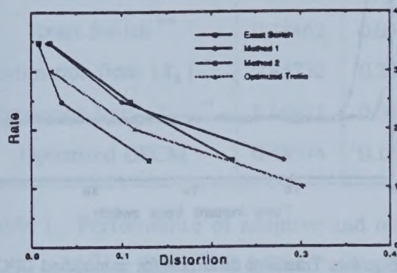


Fig. 1. The rate-distortion performance of trellis coding schemes.
$\rho^{(1)} = 0.2$, $\rho^{(2)} = 0.95$, $\delta = 0.1$.

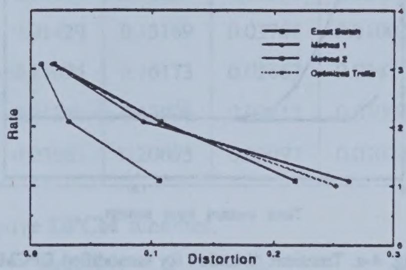


Fig. 2. The rate-distortion performance of trellis coding schemes.
$\rho^{(1)} = 0.2$, $\rho^{(2)} = 0.95$, $\delta = 0.01$.

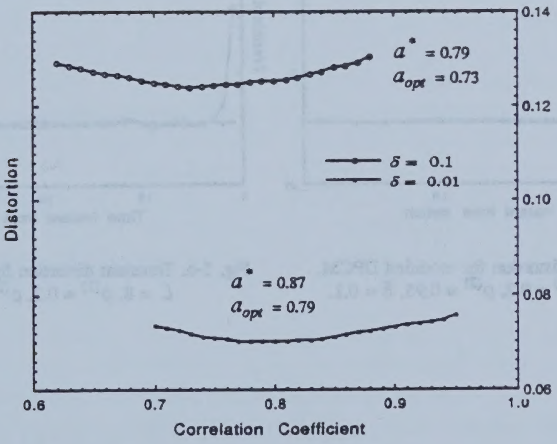


Fig. 3. Distortion vs. Prediction Coefficient for optimized DPCM.
$L = 8$, $\rho^{(1)} = 0.2$, $\rho^{(2)} = 0.95$.

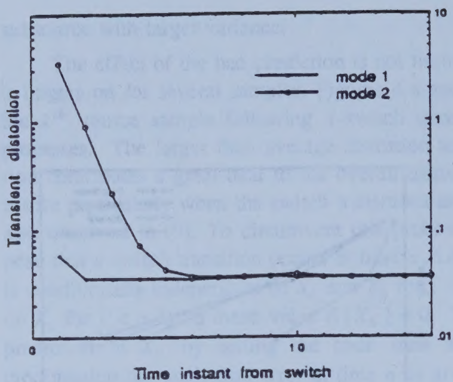


Fig. 4-a. Transient distortion for unmodified DPCM.
$L = 8, \rho^{(1)} = 0.2, \rho^{(2)} = 0.95, \delta = 0.1.$

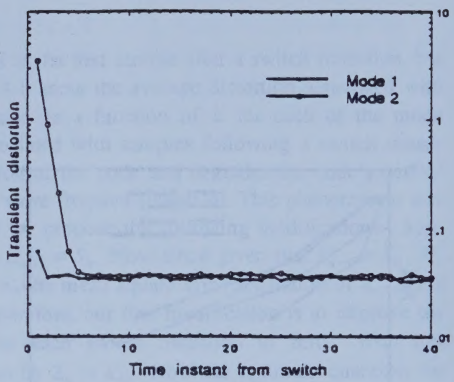


Fig. 4-b. Transient distortion for unmodified DPCM.
$L = 8, \rho^{(1)} = 0.2, \rho^{(2)} = 0.95, \delta = 0.01.$

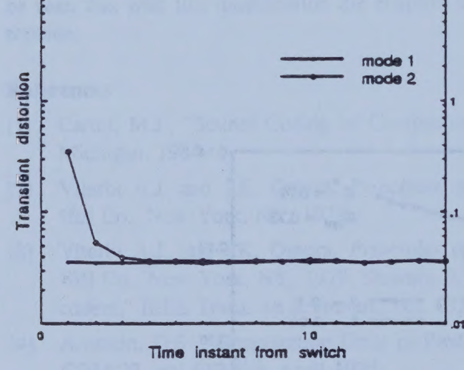


Fig. 5-a. Transient distortion for modified DPCM.
$L = 8, \rho^{(1)} = 0.2, \rho^{(2)} = 0.95, \delta = 0.1.$

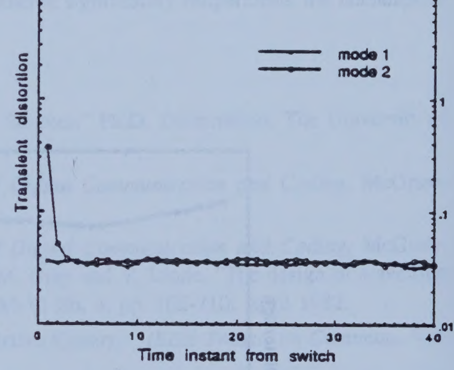


Fig. 5-b. Transient distortion for modified DPCM.
$L = 8, \rho^{(1)} = 0.2, \rho^{(2)} = 0.95, \delta = 0.01.$

	$\delta = 0.1$			$\delta = 0.01$		
System used	$L = 4$	$L = 8$	$L = 16$	$L = 4$	$L = 8$	$L = 16$
Exact Switch *	0.45870	0.23374	0.13197	0.16295	0.05898	0.02499
Exact Switch **	0.18862	0.05352	0.01429	0.13169	0.03761	0.01002
Estimation from $\{X_k\}$ *	0.45730	0.23200	0.13095	0.16173	0.05867	0.02477
Estimation from $\{X_k\}$ **	0.16931	0.04745	0.01260	0.13006	0.03717	0.01002
Optimized DPCM	0.44974	0.12400	0.03563	0.20695	0.06997	0.02074

Table 1. Performance of adaptive and nonadaptive DPCM schemes.
$\rho^{(1)} = 0.2$, $\rho^{(2)} = 0.95$, * unmodified, ** modified.

Optimal Cyclic Quantization on Closed Peano Curves

Xiaolin Wu
Department of Computer Science
University of Western Ontario
London, Ontario, Canada N6A 5B7 [*]

July 1991

Abstract – Proposed is a better performed heuristic clustering algorithm for vector quantizer design. The basic idea is to reduce the dimensionality of the problem by mapping a high-dimensional data set onto a *closed* peano curve. Then the multivariate data are classified by a globally optimal cyclic quantizer. The errors incurred by dimension reduction can be to certain extent corrected by merging the clusters formed on the closed peano curve in the original vector space.

1 Introduction

Classification of multivariate observations is a daily scientific endeavor. Many multivariate classification (clustering) techniques [1, 8] are used in practically all disciplines. These techniques fall into two major categories: 1) graph theoretic methods such as hierarchical clustering and minimum spanning tree-based clustering; and 2) partitioning methods that minimize an objective function defined on different partitions of the data set. This paper is concerned with the partitioning methods.

Problem formulation. Given a finite set $S = \{\mathbf{x}_n;\ 1 \le n \le N\}$ of N d-variate data of weights $W(\mathbf{x})$, to group the data by their similarity in attributes, we wish to partition S into K subsets (clusters), S_k, $k = 1, 2, \ldots, K, K < N$, under the criterion of minimizing the grouping cost function

$$E(S_1, S_2, \ldots, S_K) = \sum_{k=1}^{K} \sum_{\mathbf{x} \in S_k} W(\mathbf{x}) \|\mathbf{x}, \mathbf{c}_k\|, \tag{1}$$

where $\mathbf{c}_k \in \Re^d$ is the representative (cluster centroid) of S_k, and $\|\mathbf{x}, \mathbf{y}\|$ denotes an error function on the difference between $\mathbf{x}$ and $\mathbf{y}$. Unfortunately this is a formidable discrete optimization problem with an exponential search domain: $O(K^{N^2})$ possible ways of forming K groups in N objects [1] !

Previous work. Two groups of researchers have worked extensively on the clustering algorithms: those in computer science investigating the problem in computational geometry frame work [4, 5, 11, 12, 14], and those in vector quantization striving to design optimal codebooks [7, 9]. Many proofs for the NP-completeness of the optimal K-clustering problem for variable K were given for various cost functions [3, 6, 11, 14]. Although the problem is solvable in polynomial time when K is fixed [4, 5], those algorithms run in $O(N^{cK})$ time, $c > 1$, even in plane, having no practical significance. As the suboptimal substitutes, the classical K-means iterative clustering algorithm (called the LBG algorithm or generalized Lloyd's algorithm [9] by quantization researchers) and its variant ISODATE [10] algorithm are widely used.

---

[*] Supported by grants from the Natural Sciences and Engineering Research Council of Canada

The K-means iterative clustering method is a heuristic approach aiming at a local minimum of the cost function (in extremely unlucky cases it can even output a local maximum). More annoyingly, no nontrivial bound on its optimality degradation compared with the true global minimum is known. The main purpose of this paper is to provide an alternative heuristic algorithm that is based on a constrained global optimization scheme.

2 Dimension Reduction by Peano Curve

If the data under consideration are scalars, then optimal K-clustering is no longer NP-complete even when K is part of the input. In 1964 Bruce [2] proposed a dynamic programming algorithm to find the global minimum of the cost function. For mean square error, his algorithm runs in $O(KN^2)$ time. This time complexity was reduced by us to $O(KN \log N)$ [15] and very recently to $O(KN)$ [16]. Now a natural question surfaces: what makes clustering vectors so difficult?

The optimal K-partition of S corresponds to a Voronoi diagram on the centroids of the resulting K optimal clusters. It is this geometric structure that permits $O(N^{cK})$ solutions to the optimal K-clustering versus the $O(K^{N^2})$ exhaustive search algorithm. The reason for the existence of strikingly faster clustering algorithms in one-dimensional space is a special property that is enjoyed by one-dimensional Voronoi diagram but not by its higher dimensional counterparts. In one dimension, the halfplane h_i between two adjacent cluster centroids c_i and c_{i+1} is the midpoint between them, and the $K - 1$ halfplanes h_i, $1 \le i < K$, define the K Voronoi regions which are the intervals $[h_i, h_{i+1}]$. In this construction a halfplane h_i also defines a partition of Voronoi regions in the sense that all the Voronoi regions $[h_t, h_{t+1}]$ are either at the left side of h_i if $t < i$ or at the right side of h_i if $t \le i$. However, a high-dimensional Voronoi diagram does not in general have this region separability by halfplanes. This inseparability prevents the use of dynamic programming technique for optimal clustering in dimensions higher than one.

One way to optimize the K-clustering at a reasonable computational cost is to transform the data set from high-dimensional to one-dimensional space and then carry out the clustering process. The transformation should preserve the spatial coherence of the original data. To this end Hilbert's space filling curves are considered, and specifically peano curve is used. Figure 1 gives a two-dimensional and a three-dimensional peano curves. A d-dimensional peano curve imposes an order on the subspaces of a compact d-space if one orders the subspaces along the course of the curve. Consequently the peano curve scan maps a d-space onto a curve. This map has desirable properties: 1. close points on the curve must be close in the d-space; 2. close points in the d-space are likely to be close on the curve; 3. far apart points in the d-space cannot be close on the curve.

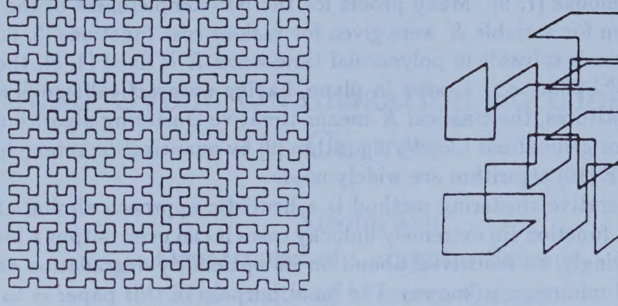


Figure 1: Examples of peano curves.

Manipulation of multivariate data using peano curves is not a new idea [13]. The original results of this paper are efficient algorithms for globally optimal K-clustering on peano curves and the use of *closed* peano curves for better preservation of spatial information.

3 Optimal K-Clustering on Peano Curve

After N points in S are mapped onto a peano curve, each point $\mathbf{x} \in S$ has a position, or rank, on the curve, denoted by $R(\mathbf{x})$. Consequently, the peano curve scan is the one-to-one map $R : S \to \{1, 2, \cdots, N\}$. Next we generalize the optimal scalar quantizer to a peano curve carrying multidimensional points. A partition of an N-point peano curve into K contiguous segments is called a $K:N$ quantizer $\mathbf{q}$, which is a partition of the ordered set $\{i \mid 1 \le i \le N\}$ into K subsets, $\{i \mid q_{j-1} < i \le q_j\}$, $0 < j \le K$. If we define a finite set:

$$Q_n^k \equiv \{\mathbf{q} \mid 0 \equiv q_0 \le q_1 < q_2 < \cdots < q_{k-1} < q_k \equiv n\} \subset \aleph^{k+1}, \tag{2}$$

where $\aleph$ is the set of all natural numbers, then there is a one-to-one map between a $\mathbf{q} \in Q_n^k$ and a possible $k:n$ quantizer. The quantization error (the same as the grouping cost as defined earlier) of a $K:N$ quantizer $\mathbf{q} \in Q_N^K$ is:

$$E(\mathbf{q}) = \sum_{j=1}^{K} \mathcal{E}(q_{j-1}, q_j] = \sum_{j=1}^{K} \sum_{q_{j-1} < R(\mathbf{x}) \le q_j} W(\mathbf{x}) \|\mathbf{x}, \mathbf{c}_j\|, \tag{3}$$

where $\mathbf{c}_j$ is the centroid of the point set $S(q_{j-1}, q_j] \equiv \{\mathbf{x} \mid q_{j-1} < R(\mathbf{x}) \le q_j\}$, and $\mathcal{E}(q_{j-1}, q_j]$ is the quantization error of $S(q_{j-1}, q_j]$. Notice that the intervals involved are open on the left but close on the right. It is important to realize that $\mathcal{E}(q_{j-1}, q_j]$ is still defined in $\Re^d$ not in $\Re$. Thus the $K:N$ quantizer formulated above is a *vector* not scalar quantizer. Now we can define the optimal $K:N$ quantizer $\hat{\mathbf{q}}$ to be the one that minimizes Eq(3), i.e., $\hat{\mathbf{q}} = \arg\min_{\mathbf{q} \in Q_N^K} E(\mathbf{q})$. Optimal $K:N$ quantization (optimal K-clustering) on a peano curve is therefore a non-linear programming problem of minimizing $E(\mathbf{q})$

subject to $\mathbf{q} \in Q_N^K$. By peano curve scan the size of problem domain is reduced from $O(K^{N^2})$ to $|Q_N^K| = \binom{N-1}{K-1} = O(N^{K-1})$ which is already polynomial for fixed K. But finding the optimum by enumeration is still intractable for modest N and K and $K \ll N$. Fortunately, the dynamic programming technique can be used to minimize Eq(3), and this is precisely the reason for conducting a peano curve scan prior to clustering process.

The first $t-1$ components of a $k:n$ quantizer $\mathbf{q} \in Q_n^k$, where $1 < t < k$, $k < n$, $k \leq K, n \leq N$, comprise a $t : q_t$ quantizer on the point set $S(q_0, q_t]$. It can be proven by contradiction that the first $t-1$ components of the optimal $K:N$ quantizer $\hat{\mathbf{q}}$ must be the optimal $t : q_t$ quantizer on $S(\hat{q}_0, \hat{q}_t]$. This property, called the principle of optimality in optimization literature, enables a simple dynamic programming algorithm for optimal $K:N$ quantization. Indeed, let L be an operator on a pair of integers, k and n, $1 < k < n$, defined by

$$\mathrm{L}[k,n] \equiv (\hat{\mathbf{q}}_n^k)_{k-1}, \ \mathrm{L}^i[k,n] \equiv \mathrm{L}[k-i+1, \mathrm{L}^{i-1}[k,n]], \ \mathrm{L}^0[k,n] \equiv n, \tag{4}$$

where the subscript of $(\hat{\mathbf{q}}_n^k)_i$ stands for the i^{th} component of the vector $\hat{\mathbf{q}}_n^k$. By the principle of optimality one can infer that $(\hat{\mathbf{q}}_N^K)_i = \mathrm{L}^{K-i}[K,N]$, $1 \leq i < K$,

$$E(\hat{\mathbf{q}}_n^k) = E(\hat{\mathbf{q}}_{\mathrm{L}[k,n]}^{k-1}) + \mathcal{E}(\mathrm{L}[k,n], n), \quad 2 \leq k < n \leq N, \tag{5}$$

and

$$\mathrm{L}[k,n] = \arg\min_{k<i<n} \{E(\hat{\mathbf{q}}_i^{k-1}) + \mathcal{E}(i,n)\}. \tag{6}$$

Hence $\mathrm{L}[k,n]$ can be determined by a linear search provided $E(\hat{\mathbf{q}}_i^{k-1})$, $k \leq i < n$, are all known. Once $\mathrm{L}[k,n]$ is determined $E(\hat{\mathbf{q}}_n^k)$ also becomes known by Eq(5). This suggests that $\hat{\mathbf{q}}_N^K$ can be constructed by bottom-up dynamic programming. First note that the errors of one-level quantizers are trivially $E(\hat{\mathbf{q}}_n^1) = \mathcal{E}(0,n)$, $1 \leq n \leq N$. Then by Eq(6) and Eq(5), $\mathrm{L}[2,n]$ and $E(\hat{\mathbf{q}}_n^2)$, $2 \leq n \leq N$, can be computed and stored as intermediate results for later use. In general, the dynamic programming process determines $\mathrm{L}[k,n]$ and $E(\hat{\mathbf{q}}_n^k)$, $k \leq n \leq N$, by referring to $E(\hat{\mathbf{q}}_n^{k-1})$, $k-1 \leq n \leq N$, and it of course remembers all the results just obtained to facilitate the computations of $\mathrm{L}[k+1,n]$ and $E(\hat{\mathbf{q}}_n^{k+1})$, $k+1 \leq n \leq N$. The process is carried out until k has been incremented from 2 up to K and $\mathrm{L}[K,N]$ is finally obtained. The parameters of optimal $K:N$ quantizer $\hat{\mathbf{q}}_N^K$ can then be obtained by the relation $(\hat{\mathbf{q}}_N^K)_{K-i} = \mathrm{L}^i[K,N] = \mathrm{L}[K-i+1, \mathrm{L}^{i-1}[K,N]]$, $i = 1, 2, \ldots, K-1$, i.e., restore $\mathrm{L}^i[K,N]$ from $\mathrm{L}^{i-1}[K,N]$. The pseudo code of the algorithm is given below.

```
Algorithm. Optimal quantization by dynamic programming.
Input:          $N, K, S, W$.
Output:         $\hat{\mathbf{q}}_N^K$.
Global arrays:  $E[k,n] \equiv E(\hat{\mathbf{q}}_n^k)$, $\mathrm{L}[k,n] \equiv (\hat{\mathbf{q}}_n^k)_{k-1}$;
Initialization: $E[1,n] := \mathcal{E}(0,n)$, $1 \leq n \leq N$; $E[k,k] := 0$, $\mathrm{L}[k,k] := k-1$, $1 \leq k \leq K$.
begin
            for $k := 2$ to $K$ do
               for $n := k+1$ to $N-K+k$ do begin
                  $cut := n-1$; $e := E[k-1,n-1]$;
                  for $t := n-2$ downto $k-1$ do
                  if $E[k-1,t] + \mathcal{E}(t,n) < e$ then begin
                     $cut := t$; $e := E[k-1,t] + \mathcal{E}(t,n)$;
            end;
            L[k, n] := cut; E[k, n] := e;
        end;
        output Lchain(K, N) as $\hat{\mathbf{q}}_N^K$;
end.

function Lchain(k, n) : q ∈ $Q_n^k$;
begin
        t := n;
        for j := k − 1 downto 1 do $q_j$ := t := L[j + 1, t];
        return(q);
end
```

In the kernel of the algorithm, the quantization errors $\mathcal{E}(l, r]$, $0 \le l < r \le n$, are evaluated. In order not to repeatedly compute $\mathcal{E}(l, r]$ for the same $(l, r]$, we can precompute $\mathcal{E}(l, r]$ for all possible pairs l and r and store them for future reference. But there are $O(N^2)$ those pairs, hence $O(N^3)$ time and $O(N^2)$ space are required by such a preprocessing. However, we can do better if $\|\mathbf{x}, \mathbf{c}\|$ is the squared d-dimensional Euclidean distance between $\mathbf{x}$ and $\mathbf{c}$, that is,

$$\begin{aligned} \mathcal{E}(l,r] &= \sum_{l<R(\mathbf{X})\le r} W(\mathbf{x})(\mathbf{x}-\mathbf{c})^T(\mathbf{x}-\mathbf{c}) \\ &= \sum_{m=1}^{d} \left\{ \sum_{l<R(\mathbf{X})\le r} W(\mathbf{x})x_m^2 - \frac{\left[\sum_{l<R(\mathbf{X})\le r} W(\mathbf{x})x_m\right]^2}{\sum_{l<R(\mathbf{X})\le r} W(\mathbf{x})} \right\} \qquad (7) \end{aligned}$$

Now define the quantities:

$$\begin{aligned} M_2(n) &\equiv \sum_{1\le R(\mathbf{X})\le n} \left(W(\mathbf{x}) \sum_{m=1}^{d} x_m^2 \right) \\ M_1(m,n) &\equiv \sum_{1\le R(\mathbf{X})\le n} W(\mathbf{x})x_m \\ M_0(n) &\equiv \sum_{1\le R(\mathbf{X})\le n} W(\mathbf{x}) \qquad (8) \end{aligned}$$

and rewrite Eq(7) as

$$\mathcal{E}(l,r] = M_2(r) - M_2(l) - \frac{\sum_{m=1}^{d} \left[W_1(m,r) - W_1(m,l)\right]^2}{M_0(r) - M_0(l)} \qquad (9)$$

Therefore, if the quantities in Eq(8) are precomputed and stored for $1 \le n \le N$, $1 \le m \le d$, $\mathcal{E}(l, r]$ can be evaluated in $O(d)$ time. Consequently, the inner t loop of our algorithm can be executed in $O(d(n - k))$ time. Thus the total computational cost of the dynamic programming process is determined by the simple counting:

$$\sum_{k=2}^{K} \sum_{n=k+1}^{N-K+k} d(n-k) = \frac{K-1}{2}[(N-K)^2 + N - K]d. \qquad (10)$$

Clearly, $O(dN)$ time suffices to precompute and save the quantities $M_2(n)$, $M_1(m,n)$ and $M_0(n)$, $1 \le n \le N$, $1 \le m \le d$, and $O(K)$ time is taken by the function Lchain(K,N), being both insignificant compared with Eq(10). Hence the time complexity of optimal $K\!:\!N$ quantization by dynamic programming is $O(dKN^2)$ when $K \ll N$.

The significance of the development of this section is to show that by restricting the K-partition of S to K contiguous segments on a peano curve, optimal multivariate K-clustering becomes computationally feasible.

4 Closed Peano Curves

By joining the two ends of a peano curve we get a closed curve. The closed peano curve preserves the spatial information better than the open one because the maximum point-to-point distance on the former is only a half of that of the latter. The advantage of the closed peano curve over the open one is clearly demonstrated by Figure 2. For instances, group a is separated on the open peano curve but remains adjacent on the closed peano curve; group d is concentrated much more on the closed curve than on the open one.

Closed peano curves are no more difficult than the open ones to be generated. A closed peano curve can be easily described by an L-system, and the algorithm, although fun, is omitted to save space.

5 K-Clustering by Optimal Cyclic Quantizer

In this section we will sketch an algorithm, called optimal cyclic quantizer, for globally optimal K-clustering on a closed peano curve, which outperforms the the previous algorithm on an open peano curve. A cyclic $K:N$ quantizer generates a partition of a closed N-point peano curve into K contiguous segments.

The N points on the closed peano curve form a ring. If we cut this ring at some point we obtain a linear sequence of those points. All together there are N different such sequences. To enumerate all these sequences, we take an arbitrary one ν and assign the rank $R(\mathbf{x}) = i$ to an $\mathbf{x} \in S$ that is at the i^th position in the chosen sequence ν. Then we concatenate ν to itself and form a sequence ω of length $2N$. Clearly, in the new sequence ω, $R^{-1}(i) = R^{-1}(j)$ if and only if $|j-i| = N$, $1 \le i,j \le 2N$, where $R^{-1}(i)$ is the point $\mathbf{x}$ whose rank in the sequence is i. Similarly to our discussion about the $K\!:\!N$ quantization on an open peano curve, we define finite sets

$$Q_N^K(i) \equiv \{\mathbf{q} \mid i \equiv q_0 \le q_1 < q_2 < \cdots < q_{K-1} < q_K \equiv N+i\} \subset \aleph^{K+1}, \quad 0 \le i < N. \qquad (11)$$

Then given an i, $Q_N^K(i)$ is the set of all possible $K\!:\!N$ quantizers on an open peano curve broken from the closed one. Our goal is to compute the optimal $K\!:\!N$ cyclic quantizer

$$\dot{\mathbf{q}} = \arg \min_{\mathbf{q} \in \cup_{i=0}^{N-1} Q_N^K(i)} E(\mathbf{q}) \qquad (12)$$

A straightforward solution will be to compute $\hat{\mathbf{q}} \in Q_N^K(i)$ for all $0 \le i < N$, and then let $\dot{\mathbf{q}}$ be the one of the minimum error. Since for fixed i finding the optimal quantizer in $Q_N^K(i)$ takes $O(KN^2)$ time, $O(KN^3)$ time is needed to find $\dot{\mathbf{q}}$. But a further reflection can lead to an algorithm of $O(N^3)$ expected time complexity. Let $\mathbf{q}^* \in Q_N^K(i)$ be the

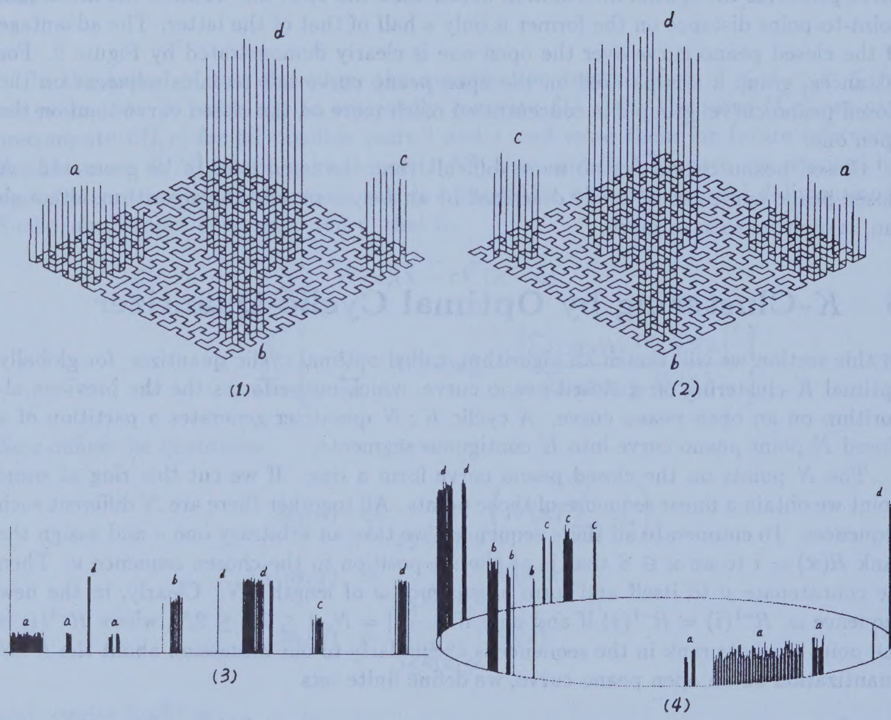

d
a
c
b
(1)
d
c
a
b
(2)
a
a
a
d
b
d
d
c
d
(3)
d
d
b
b
b
c
c
c
d
a
a
a
(4)

Figure 2: Open and closed peano curve scans of a two-dimensional histogram: (1) and (2), and their expansions into a line and a closed curve respectively: (3) and (4).

optimal $K : N$ quantizer on the open peano curve for i. Then all the optimal $K : N$ quantizers on the open peano curves generated by breaking the closed peano curve at positions q_j^*, $1 \leq j \leq K$, correspond to the same partition of the closed peano curve. Therefore, once $\mathbf{q}^* \in Q_N^K(i)$ is known we can ignore the sets $Q_N^K(q_j^*)$, $1 \leq j \leq K$, in the search for $\dot{\mathbf{q}}$. On average for a cost of $O(KN^2)$ to compute an optimal quantizer on an open peano curve, we can eliminate $O(K)$ sets of $Q_N^K(i)$ for as the candidates for the optimal cyclic quantizer $\dot{\mathbf{q}}$. Since there are N sets of $Q_N^K(i)$ to be searched we need to compute $O(N/K)$ of optimal quantizers for open peano curves to complete the search for $\dot{\mathbf{q}}$ in the expected case, verifying our claim for $O(N^3)$ expected time complexity. The details of this algorithm are dropped due to the space limitation.

6 Improvement by Merge

Far apart points in the high-dimensional space cannot be close together on the peano curve, so our algorithms cannot produce false clusters. However, close points in a high-dimensional space could be mapped to far apart positions on the peano curve. This means that a valid cluster could be split by the peano curve scan. To improve clustering validity one should generate more intermediate clusters on the peano curve than originally intended by the proposed optimal K-clustering process, and then merge those intermediate clusters until the desired number of clusters are formed. Note that this merging operation is in essence another clustering process, but with the size of input point set greatly reduced by first clustering on the closed peano curve. Nevertheless, the globally optimal merge is still an NP-complete problem. Instead a greedy merge algorithm was used in our practice. We consider the merge of a pair of clusters at a time. Among all the possible merges we always merge the two clusters whose union yields the least increase in the grouping cost until only K clusters are left.

The two-step process of splitting by optimal cyclic quantizer plus greedy merging performs better than pure greedy merge from the point level. This is because optimal cyclic quantization allows a global optimization, although restricted to the peano curve, while greedy merge cannot.

The new clustering algorithms on both open and closed peano curves were tested on simulated two-dimensional data sets. The optimal cyclic quantizer on a closed peano curve achieved 5% to 20% smaller total mean squared error than the optimal quantizer on an open peano curve. It was also observed that the new algorithms generated better clusters than the conventional K-means method of random initialization in terms of smaller grouping costs.

References

[1] M. R. Anderberg, *Cluster Analysis for Applications*, Academic Press, New York, 1973.

[2] J. D. Bruce, "*Optimum Quantization*," Sc. D. thesis, M.I.T., May 14, 1964.

[3] P. Brucker, "On the complexity of clustering problems" in *Optimization and Operations Research* (R. Henn, B. Korte, and W. Oettli, eds.), pp. 45-54, Springer-Verlag, 1977.

[4] V. Capoyleas, G. Rote and G. Woeginger, "Geometric clustering," *J. Algorithm*, vol. 12, pp. 341-356, 1991.

[5] Z. Drezner, "The p-center problem - heuristic and optimal algorithms," *J. Oper. Res. Soc.*, vol. 35, pp. 741-748, 1984.

[6] M. R. Garey, D. S. Johnson, H. S. Witsenhausen, "The complexity of the generalized lloyd-max problem," *IEEE Trans. Inf. Theory*, vol. IT-28, pp. 255-256, March 1982.

[7] R. M. Gray, "Vector quantization," *IEEE ASSP Mag.*, pp. 4-29, April 1984.

[8] J. A. Hartigan, *Clustering algorithms*, Wiley, New York, 1975.

[9] Y. Linde, A. Buzo, R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. COM-28, pp. 84-95, Jan. 1980.

[10] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations", *Proc. of Fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, p. 281-297, 1967.

[11] N. Megiddo and K. J. Supowit, "On the complexity of some common geometric location problems," *SIAM J. Comput.* vol. 13, pp. 182-196, 1984.

[12] N. Megiddo and A. Tamir, "On the complexity of locating linear facilities in the plane," *Oper. Res. Lett.* vol. 1, pp. 194-197, 1982.

[13] R. J. Stevens, A. F. Lehar, and F. H. Preston, "Manipulation and presentation of multidimensional image data using the peano scan," *IEEE Trans. Pat. Ana. Mach. Intel.*, vol. PAMI-5, pp. 520-526, 1983.

[14] K. J. Supowit, "Topics in computational geometry," Ph.D. thesis, D. of Computer Science, U. of Illinois at Urbana-Champaign, 1981.

[15] X. Wu, and J. Rokne, "An $O(KN \lg N)$ algorithm for optimum K-level quantization on histogram of N points," *Proc. ACM 1989 Conference of Computer Science*, pp. 339-343, Feb. 1989.

[16] X. Wu, "Discrete optimal quantization by matrix-searching," *J. of Algorithms* (in press).

Unequal Error Protection (UEP) Codes Applied to Transmission of Analog Sources

H. Arda Aksu and M. Salehi
Northeastern University
Electrical and Computer Engineering Department

Unequal Error Protection Codes have the property that their error correction capability is described in terms of correcting errors in specific digits of a code word even though other digits in the code word may be decoded incorrectly. In [1] first the concept of unequal error protection (UEP) was introduced. They introduced a so-called separation vector to measure the error-correcting capability of a liner UEP code. To each digit of the code word was assigned a separation value s_i. Then nearest neighbor decoding guarantees the correct interpretation of the i'th code word digit if no more than $\lfloor (s_i - 1)/2 \rfloor$ errors have occured in the transmitted code word. In [2] Van Gils compiled a table for UEP capabilities of binary cyclic codes of odd lengths up to 39. In [3] the table was extended to lengths up to 65. UEP Codes may find application in numerical data-transmission systems, where the digits of the information sequence can be of differing significance.

In order to provide higher error protection for some digits in a code word one must make the columns of the H matrix corresponding to those digits be of "higher linear independence" than the other columns of H.In [1] Masnick & Wolf proposed a method -H matrix overlapping- for accomplishing this. On the other hand, by their nature, cyclic systematic codes cannot provide UEP. In [2] , a general method for constructing binary nonsystematic cyclic UEP codes, which are majority logic decodable, was provided. Later Van Gils computed separation vectors of some cyclic codes by comparing the weight distributions of its cyclic subcodes. This was achieved with an ordering $M_1, M_2, \ldots, M_v$ of generator matrices of v minimal ideals where the direct sum of these minimal ideals form a cyclic code. Performance of binary UEP codes seem to depend very much on the application. Although *cyclic* UEP codes don't compare favorably with the best known BCH codes, they provide further protection for some digits than guaranteed by the minimum distance of the code (d_{min}) for the class of majority logic decodable codes. And general class of UEP codes can provide less distortion than equal error protection codes.

In this work bounds on the expected value of distortion (due to incorrect transmission) are derived and different aspects of the problem are discussed. In a linear (n, k) code with separation vector $\mathrm{s} = (s_1, s_2, \ldots, s_k)$, we can associate each input message digit i with a distortion measure Δ_i, which will reflect the penalty (cost) paid by decoding the i'th digit incorrectly. By defining a random variable e, which is equal to the number of errors occurred in the input message during transmission, the expected value of distortion can be written in the following form:

$$E(D) = \sum_{i=0}^{k} P(t_i \leq e \leq t_{i+1}) \sum_{j=0}^{i} \Delta_j$$

where $t_i = \left\lfloor \frac{s_i - 1}{2} \right\rfloor$

Also it can be shown that

$$E(D) = \sum_{j=1}^{k} \Delta_j - \sum_{i=1}^{k} \Delta_i \, P(e \leq t_i)$$

and by using the Chernoff Bound, a lower bound for distortion is obtained as:

$$E(D) \geq \sum_{j=1}^{k} \Delta_j - \sum_{i=1}^{k} \Delta_i \, e^{-n[h(t_i/n) - (t_i/n)\ln p - (1-t_i/n)\ln(1-p)]}$$

where p is the channel transmission error probability and $h(.)$ is the binary entropy function. Further simplification is possible by assuming structure on Δ_i's (for example uniform scalar quantization, nonuniform quantization and adaptive quantization schemes with mean square error distortion measure). A numerical evaluation of the above expression for a specific code with two levels of error protection yields to the fact that lower distortion values can be attained by unequal error protection.

In [1] Masnick & Wolf derived a recursive relationship for the number of correctable error patterns of weight $i\,(A_i)$ in terms of $A_{i-1}, A_{i-2}, \ldots, A_1$. But, in contrast with what one would expect, they considered error protection of single positions in code words rather than the positions in the input data block. With some modification, a similar recursion is obtained for the number of correctable errors in the input data block and brought into scenario in order to achieve a tighter bound for the expected value of distortion. Subtracting the number of correctable error patterns, one obtains:

$$E(D) = \sum_{i=0}^{k} \left(\sum_{j=t_i+1}^{t_{i+1}} \left(\binom{n}{j} - A_j\right) p^j (1-p)^{n-j} \sum_{l=0}^{i} \Delta_l \right)$$

where

$$A_i = \sum_{j=1}^{i} \binom{T_i + j - 1}{j} (-1)^{1+j} A_{i-j}$$

$T_i =$ the number of code digits that are i or more protected
$A_0 = 1$

Using the recursive relation for A_j's, tighter bounds are obtained on $E(D)$. Based on this bound, one can find bounds on t_i's (or equivalently the separation vector $\mathbf{s}$) that can achieve a certain level of distortion. These bounds together with bounds on n for a given separation vector obtained by Van Gils, give general bounds on the rate of the UEP code which is capable of achieving a given distortion.

References

[1] B. Masnick and J. Wolf, "On linear unequal error protection codes," *IEEE Trans. Inform. Theory,* vol. 13, pp. 600-607, Oct. 1967

[2] Wil J. Van Gils, "Two topics on Linear Unequal Error Protection Codes: Bounds on Their Length and Cyclic Code Classes," *IEEE Trans. Inform. Theory,* Vol. IT-29, No. 6, Nov. 1983

[3] Mao-Chao Lin, Chi-Chang Lin and Shu Lin, "Computer Search for Binary Cyclic UEP Codes of Odd Length Up to 65," *IEEE Trans. Inform. Theory,* Vol. 36, No. 4, July 1990.

The Analysis of the C-Chain Code

Yu-Wei Chen Samuel C. Lee

School of EECS
University of Oklahoma
Norman, Oklahoma 73019

This paper presents a new chain code for encoding three-dimensional (3D) curves, called the C-chain code. Special features of this chain code are described. It is shown that the C-chain code provides a wider range of variations in link length than the Freeman chain code.

The average quantization error of the C-chain code is computed. It is shown that the quantization error is directly proportional to the fineness of the quantization cube. Based on the probabilities (frequencies) of occurrences of the quantization points of the C-chain codes, it is further shown that the selection of the orientation of the next quantization cube by the C-chain coding rule described in this paper is optimal.

I. Introduction

In the past, there has been intensive research in 2-D chain codes for line drawings. Groen and Verbeek [1] have estimated the probabilities of odd or even 2-D Freeman codes. In Freeman and Saghri's works [2][3], they have developed a procedure for computing the relative probabilities of the generalized chain code and showed that the average quantization error is independent of the form of the code. Later Minami and Shinohara [4] proposed a scheme called the multiple grid chain code and showed its performance is better than the previously published ones in terms of the encoding efficiency, naturalness of the encoded lines, and the rate distortion measure.

However, there was little literature about 3-D chain codes. Recently we suggested a new coding scheme for 3-D chain coding, called the C-chain code. This code is similar to the multiple chain code which places its quantization points at the centers of subsurfaces of the side surfaces of the quantization cube, not at its corners like those used by Freeman [5]. The organization of the paper is as follows. The C-chain code will be described briefly in section II. In section III, we shall define a average distance (error) function of two pieces of directed curves, then calculate the average errors of the C-chain code basing on this function. In section IV, the frequency estimation of occurrences of the quantization points will be presented. A conclusion is given in section V.

II The C-chain code

Referring to Figure 1(b), six square cubes C_0, C_1, C_2, C_3, C_4, and C_5, with edge length 4q are used for quantization of space curves. The labeling of quantization points of the six surfaces of a quantization cube is shown in Figure 1(a). Each surface of a cube is divided into four equal subsurfaces, and the quantization points are set at the centers of these subsurfaces. A curve crossing two subsurfaces of the cube is approximated by a straight line called a link, which is the line connecting the two quantization points on these two subsurfaces. The first quantization point on the subsurface where the curve enters to the cube is labeled as '00.'

To encode a curve, the '00' point of the '00' subsurface is superimposed with the starting point of the curve to be encoded. The first quantization point is determined by the quantization point of the subsurface that the curve passes through. This quantization point then becomes the new '00' for finding the next quantization point. The selection of the next quantization cube, its orientation, and origin is determined by the following manner: if the quantization point is on the i surface, then the next quantization cube will be cube (5-i). After the next quantization cube has been decided, rotate the new quantization cube to match the relative orientation of the current quantization cube, and let the quantization point just found coincide the point '00' of the new quantization cube. For example, in Figure 2 the left cube is the current cube, and assume the curve will passes through subsurface '22'. The next cube will be C_3 (5-2=3), and the X-, Y-, and Z-axes of next cube are parallel to Z-, X-, and Y-axes of the current cube, respectively. It will be shown that when these quantization orientations are used to encoding the space curve, the link lengths will be longest.

III. The average error between a curve segment and its code link

First, we define the average distance measure of two directed curves. Let C_1 and C_2 be two pieces of directed curves and p_0, p_n and p_0', p_n' the end-points of those two curves, respectively. The parametrial expressions of those two curves are

$C_1 = (x_1(t), y_1(t), z_1(t))$ and $C_2 = (x_2(t), y_2(t), z_2(t))$, (1)

where

$$\begin{aligned} &0 \le t \le 1, \\ &p_0 = C_1\ (\ x_1(0),\ y_1(0),\ z_1(0)\ ), \\ &p_n = C_1\ (\ x_1(1),\ y_1(1),\ z_1(1)\ ), \\ &p_0' = C_2\ (\ x_2(0),\ y_2(0),\ z_2(0)\ ), \text{ and} \\ &p_n' = C_2\ (\ x_2(1),\ y_2(1),\ z_2(1)\ ). \end{aligned}$$

The average distance measure of two pieces of directed curves of C_1 and C_2 is defined as follows.

$$dis(C_1,C_2) = \int_0^1 \sqrt{(x_1(t)-x_2(t))^2+(y_1(t)-y_2(t))^2+(z_1(t)-z_2(t))^2}\ dt$$

Now we apply the above definition to calculate the average distance measure (quantization error) of a curve and its representative link. In general, the size of the quantization cube should be just fine enough such that within the region of a cube the curve can be represented as a straight line. Therefore, from now on we will restrict ourselves to assume the curve to be quantized as a straight line segment. The vector valued functions of the curve and its quantized link can be represented as

$$C:\ \begin{aligned} x &= x_0 + (x_n-x_0)\cdot t \\ y &= y_0 + (y_n-y_0)\cdot t \\ z &= z_0 + (z_n-z_0)\cdot t \end{aligned} \qquad\qquad L:\ \begin{aligned} x &= x_0' + (x_n'-x_0')\cdot t \\ y &= y_0' + (y_n'-y_0')\cdot t \\ z &= z_0' + (z_n'-z_0')\cdot t \end{aligned}$$

where $0 \le t \le 1$. (3)

The formula of the average quantization error of a curve and its representative link now becomes

$$dis(L,C) = \int_0^1 \sqrt{a\cdot t^2+b\cdot t+c}\ dt$$

where

$$\begin{aligned} a &= (x_n-x_0-x_n'+x_0')^2 + (y_n-y_0-y_n'+y_0')^2 + (z_n-z_0-z_n'+z_0')^2, \\ b &= 2(x_0-x_0')(x_n-x_0-x_n'+x_0') + 2(y_0-y_0')(y_n-y_0-y_n'+y_0') + 2(z_0-z_0')(z_n-z_0-z_n'+z_0'), \\ c &= (x_0-x_0')^2 + (y_0-y_0')^2 + (z_0-z_0')^2. \end{aligned} \qquad (4)$$

It is reasonable to assume that points of subsurface '00' are uniformly distributed upon subsurface '00' and the orientations of curves within the possible angles passing through subsurface 'XY' are equally like. The average quantization error of link 'XY' can be expressed as

$$D('XY') = \int_0^{2q}\int_0^{2q}\int_{\alpha_{min}}^{\alpha_{max}}\int_{\beta_{min}}^{\beta_{max}} dis(C,L_{XY})d\beta d\alpha dx dz \;/\; \int_0^{2q}\int_0^{2q}\int_{\alpha_{min}}^{\alpha_{max}}\int_{\beta_{min}}^{\beta_{max}} d\beta d\alpha dx dz$$

Take link '51' as an example, the average quantization error of link '51' is

$$D('51') = \int_0^{2q}\int_0^{2q}\int_{\alpha_{min}}^{\alpha_{max}}\int_{\beta_{min}}^{\beta_{max}} dis(C,L_{50})d\beta d\alpha dx dz \;/\; \int_0^{2q}\int_0^{2q}\int_{\alpha_{min}}^{\alpha_{max}}\int_{\beta_{min}}^{\beta_{max}} d\beta d\alpha dx dz \quad (6)$$

$$\approx \quad 0.5787$$

where $\alpha_{max} = \arccos((2q-x)/\sqrt{(2q-x)^2+(4q)^2})$

$$\alpha_{min} = \arccos((0-x)/\sqrt{(0-x)^2+(4q)^2})$$

$$\beta_{max} = \arcsin((2q-z) \;/\; \sqrt{(4q\cos(\alpha)/\sin(\alpha))^2+(4q)^2+(2q-z)^2})$$

$$\beta_{min} = \arcsin((0-z) \;/\; \sqrt{(4q\cos(\alpha)/\sin(\alpha))^2+(4q)^2+(0-z)^2})$$

The approximated result is obtained by applying the numerical method to the above multiple integral (6). The other results are listed as follows.

link	error	link	error	link	error	link	error
10	0.5829 q	11	0.6436 q	12	0.6036 q	13	0.5781 q
20	0.5829 q	21	0.5781 q	22	0.6036 q	23	0.6436 q
30	0.6151 q	31	0.9600 q	32	0.9128 q	33	0.6021 q
40	0.6151 q	41	0.6021 q	42	0.9128 q	43	0.9600 q
50	0.5748 q	51	0.5787 q	52	0.5739 q	53	0.5698 q

We note that the quantization errors are proportional to the fineness (q) of the quantization cube, and the difference between those average quantization errors are not significant except four links, $L_{'31'}$, $L_{'32'}$, $L_{'43'}$, and $L_{'42'}$.

IV. The probabilities (frequencies) of occurrences of the quantization points

In this section, the estimating probabilities of the occurrences of the 24-point C-chain code is presented.

Let A and B be two subsurfaces of two sequent quantization cubes. Assume that lines starting from a point of ’00’ subsurface first passes through subsurface A then crossing subsurface B. Let S_A and S_B be the two surface areas of unit spheres covering these two subsurfaces A and B, respectively (see Figure 4). The probability of the occurrence of such lines is defined by

$$p = \frac{S_B}{S_A}$$

The area S of the spherical polygon form by S_A and S_B is

$$S = \sum_{i=1}^{n} \alpha_i - (n-2)\pi$$

where n and α_i are the number and the angle of the spherical polygon, respectively. (see Figure 3)

Applying the above formula, the conditional probability measure *dp* of straight lines pass subsurfaces A and B can be derived as

$$dp = \frac{S_B}{S_A}\,dxdy = \frac{\sum_{i=1}^{n_B} \alpha_i - (n_B-2)\cdot\pi}{\sum_{i=1}^{n_A} \alpha_i - (n_A-2)\cdot\pi}\,dxdy$$

By integrating *dp* over the subsurface ’00’, the total probability measure can then be obtained, which is

$$Q = \int_0^{2q}\int_0^{2q} \frac{\sum_{i=1}^{n_B} \alpha_i - (n_B-2)\cdot\pi}{\sum_{i=1}^{n_A} \alpha_i - (n_A-2)\cdot\pi}\,dxdy$$

The calculated state transition matrix is shown in Table I, where P_{AB} is the probability of lines from the subsurface ’00’ passing the subsurfaces A and B. If we delete the first 4 rows and first 4 columns of this matrix, since they are unreachable states, this results in a Markov transition matrix. Denote it by P. If we assume that the current code after n steps does not depend on the initial code, then the stationary distribution can be found as

$$\lim_{n\to\infty} P^{(n)} = \lim_{n\to\infty} P^{(0)}P^{(n)} = P^{(0)}\, I\, \Pi = \Pi$$

where $\Pi = (\pi, \pi, \bullet \bullet \bullet, \pi)$.

In our example, π may be obtained from the matrix of Table I with first 4 rows and first 4 columns deleted, and the result is

$$\begin{aligned}\pi^t = (\ &0.004, 0.004, 0.178, 0.091,\\ &0.004, 0.091, 0.178, 0.004,\\ &0.001, 0.004, 0.001, 0.002,\\ &0.001, 0.002, 0.001, 0.004,\\ &0.066, 0.012, 0.066, 0.286\ )\end{aligned}$$

Arranging the quantization points in the descending order of their occurrence probabilities, we obtain the follows

Probability	code(link length)
0.286	53 ($2\sqrt{6}$)
0.178	12 ($\sqrt{22}$), 22 ($\sqrt{22}$)
0.091	13 ($3\sqrt{2}$), 21 ($3\sqrt{2}$)
0.066	50 ($2\sqrt{5}$), 52 ($2\sqrt{5}$)
0.012	51 (4)
0.004	10 ($\sqrt{10}$), 11 ($\sqrt{14}$), 20 ($\sqrt{10}$), 23 ($\sqrt{14}$), 31 ($\sqrt{2}$), 43 ($\sqrt{2}$)
0.002	33 ($3\sqrt{2}$), 41 ($3\sqrt{2}$)
0.001	30 ($\sqrt{10}$), 32 ($\sqrt{6}$), 40 ($\sqrt{10}$), 42 ($\sqrt{6}$)
0.000	00 (0), 01 (2), 02 ($2\sqrt{2}$), 03 (2)

From the above result, we can get a clear picture that the links have longer lengths tend to have more frequency of occurrences. Thus, the Huffman code can be applied to the C-chain code to obtain the statistical optimal coding.

V. Conclusion

In this paper, we proposed a distance measure between two directed curve segments, by using this measure the average quantization errors of the C-chain code have been calculated. The average error of each link of the C-chain code has been found to be directly proportional to the fineness of the quantization cube. The frequencies of occurrences of quantization points have also been estimated theoretically. It has been shown that the longer links tend to have higher frequencies of occurrences and can be used to justify that our selection of the next quantization cube and its orientation is optimal.

References

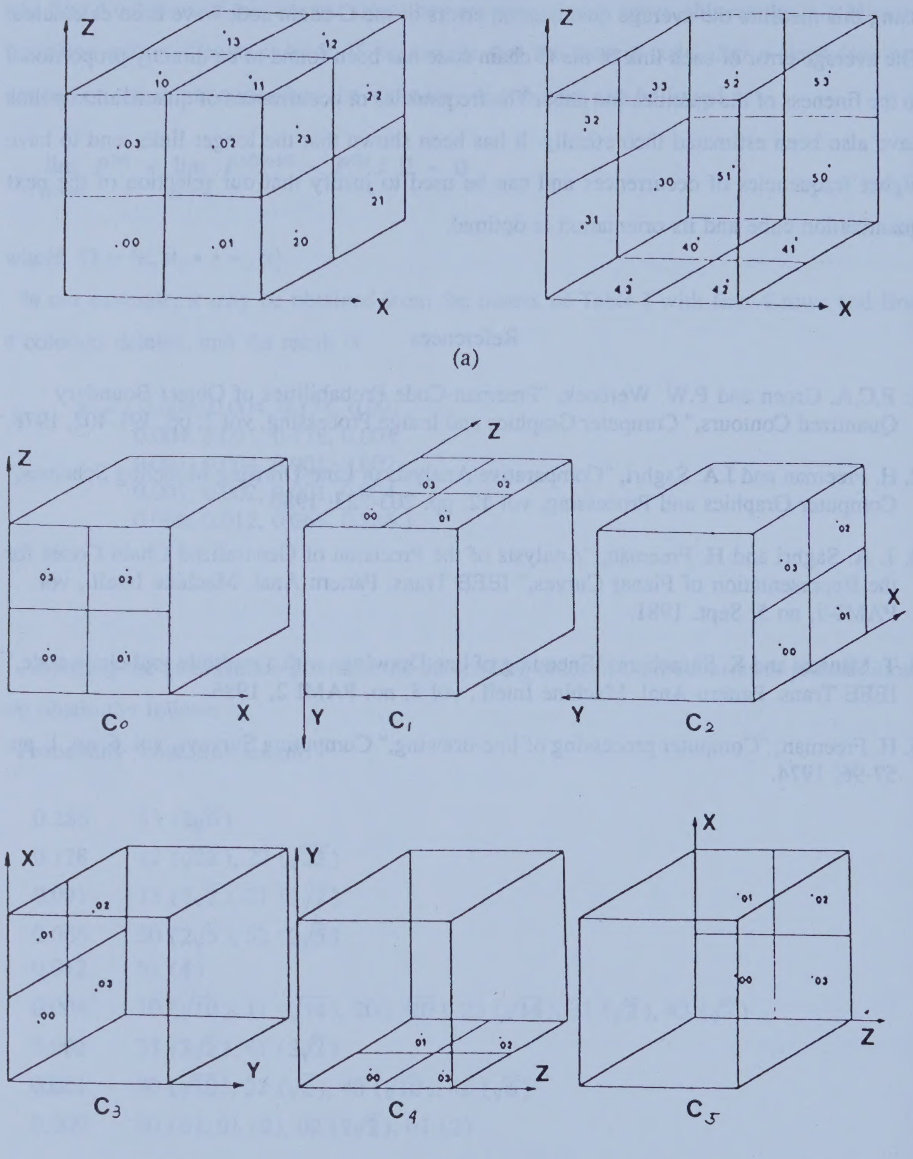
1. F.C.A. Groen and P.W. Werbeek, "Freeman-Code Probabilities of Object Boundary Quantized Contours," Computer Graphics and Image Processing, vol 7, pp. 391-402, 1978.

2. H. Freeman and J.A. Saghri, "Comparative Analysis of Line-Drawing Modeling Schemes," Computer Graphics and Processing, vol 12, pp. 203-223, 1980.

3. J. A. Saghri and H. Freeman, "Analysis of the Precision of Generalized Chain Codes for the Representation of Planar Curves," IEEE Trans. Pattern Anal. Machine Intell., vol PAMI-3, no 5, Sept. 1981.

4. T. Minami and K. Shinohara, "Encoding of line Drawings with a multiple grid chain code," IEEE Trans. Pattern Anal. Machine Intell., vol 8, no. PAMI 2, 1986.

5. H. Freeman, "Computer processing of line-drawing," Computing Surveys, vol. 6, no. 1, pp. 57-96, 1974.


Z
X
13
12
10
11
22
.03
.02
23
21
.00
.01
20
Z
33
52
53
32
51
50
30
31
40
41
43
42
X
Z
03
02
00
01
X
C0
Z
03
02
00
01
Y
C1
02
03
X
01
00
Y
C2
X
02
01
03
00
Y
C3
Y
01
02
00
03
Z
C4
X
01
02
00
03
Z
C5

(a)

(b)

Figure 1. (a) C-chain code
(b) The six quantization cubes

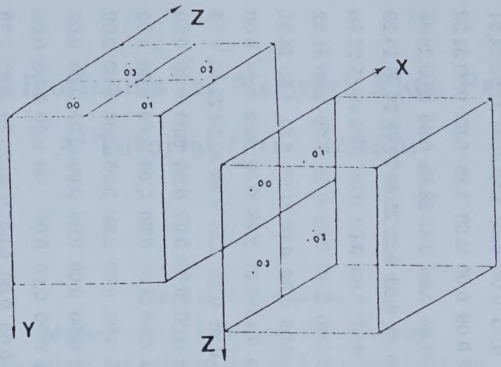

Z
X
00
01
02
03
00
01
02
03
Y
Z

Figure 2.

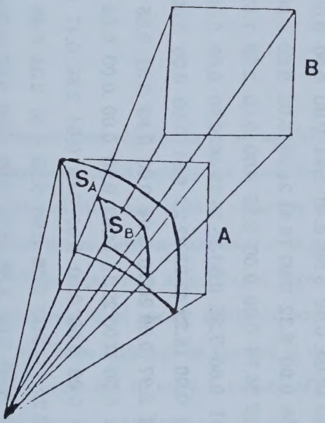


Figure 3.

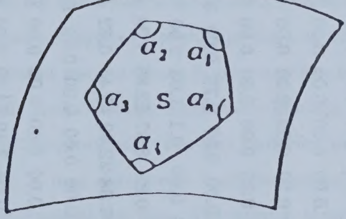

a_2
a_1
a_3
S
a_n
a_i

Figure 4.

	00	01	02	03	10	11	12	13	20	21	22	23	30	31	32	33	40	41	42	43	50	51	52	53
00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.87	0.79	0.00	0.00	0.00	0.00	0.00	1.01	0.56	0.42	2.75	20.65	27.78	25.88	19.29
11	0.00	0.00	0.00	0.00	0.00	0.00	14.84	23.63	0.00	0.1	1.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.53	2.01	31.90	24.91
12	0.00	0.00	0.00	0.00	0.00	0.00	24.99	7.40	0.89	8.37	23.85	1.34	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.85	0.00	0.00	31.32
13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.38	22.49	15.44	0.62	0.00	0.00	0.00	0.00	0.33	0.96	0.86	2.43	28.65	0.00	0.00	26.86
20	0.00	0.00	0.00	0.00	0.00	0.00	0.79	0.87	0.00	0.00	0.00	0.00	1.01	2.75	0.42	0.56	0.00	0.00	0.00	0.00	25.88	27.78	20.65	19.29
21	0.00	0.00	0.00	0.00	1.38	0.62	15.44	22.49	0.00	0.00	0.00	0.00	0.33	2.43	0.86	0.96	0.00	0.00	0.00	0.00	0.00	0.00	28.65	26.86
22	0.00	0.00	0.00	0.00	0.89	1.34	23.85	8.37	0.00	7.40	24.99	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.85	31.32
23	0.00	0.00	0.00	0.00	0.00	0.00	1.08	0.1	0.00	23.63	14.84	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	31.90	2.01	1.53	24.91
30	0.00	0.00	0.00	0.00	0.00	0.00	8.85	0.00	21.16	24.76	25.32	14.91	0.00	0.00	0.00	0.00	0.00	0.45	1.67	2.89	0.00	0.00	0.00	0.00
31	0.00	0.00	0.00	0.00	0.00	0.00	6.04	20.70	0.53	3.97	4.14	0.49	0.00	0.00	0.00	0.00	0.89	0.28	0.35	3.31	6.92	20.79	23.80	7.79
32	0.00	0.00	0.00	0.00	18.22	9.61	22.22	42.03	0.09	0.00	6.40	1.44	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
33	0.00	0.00	0.00	0.00	7.36	25.46	15.16	0.00	8.32	0.00	19.41	24.29	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
40	0.00	0.00	0.00	0.00	21.16	14.91	25.32	24.76	0.00	0.00	8.85	0.00	0.00	2.89	1.67	0.45	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
41	0.00	0.00	0.00	0.00	8.32	24.29	19.41	0.00	7.36	0.00	15.16	25.46	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
42	0.00	0.00	0.00	0.00	0.09	1.44	6.40	0.00	18.22	42.03	22.22	9.61	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
43	0.00	0.00	0.00	0.00	0.53	0.49	4.14	3.97	0.00	20.70	6.04	0.00	0.89	3.31	0.35	0.28	0.00	0.00	0.00	0.00	23.80	20.79	6.92	7.79
50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	22.15	14.62	0.00	0.00	0.00	0.00	0.00	0.40	0.93	0.86	2.34	27.91	3.12	2.78	24.89
51	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.94	2.66	0.47	0.60	0.94	0.60	0.47	2.66	22.59	25.36	22.59	20.13
52	0.00	0.00	0.00	0.00	0.00	0.00	14.62	22.15	0.00	0.00	0.00	0.00	0.40	2.34	0.86	0.93	0.00	0.00	0.00	0.00	2.78	3.12	27.91	24.89
53	0.00	0.00	0.00	0.00	0.00	0.00	22.74	8.66	0.00	8.66	22.74	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	3.35	0.37	3.35	30.14

Table I. The transition matrix of probabilities of occurances of quantization points

Auction Algorithms for Network Flow Problems:

A Tutorial Introduction

by

Dimitri P. Bertsekas

Massachusetts Institute of Technology
Laboratoty for Information and Decision Systems
Cambridge, MA 02139

Abstract

The auction algorithm is an intuitive method for solving the classical assignment problem. It outperforms substantially its main competitors for important types of problems, both in theory and in practice, and is also naturally well suited for parallel computation. It operates like a real auction where persons compete for objects by raising their prices through competitive bidding. The algorithm represents a significant departure from the cost improvement idea that underlies primal simplex and dual ascent methods; at any one iteration, it may deteriorate both the primal and the dual cost, although in the end it does find an optimal primal solution. The auction algorithm can be extended to solve a variety of other linear network flow problems such as transportation, transhipment, max-flow, and shortest path. This paper provides a tutorial introduction to the basic auction algorithm and surveys its various extensions; for textbooks presentations, see [BeT89] and [Ber91a].

1. THE ASSIGNMENT PROBLEM

The classical algorithms for solving linear network flow problems are primal cost improvement methods, including simplex methods, which iteratively improve the primal cost by moving flow around simple cycles, and dual ascent methods, which iteratively improve the dual cost by changing the prices of a subset of nodes by equal amounts. These methods are dual to each other when viewed in the context of Rockafellar's monotropic programming theory [Roc84]; they are based on cost improvement along elementary directions of the circulation space (in the primal case) or the differential space (in the dual case).

Auction algorithms are fundamentally different; they depart from the cost improvement idea and at any one iteration, they may deteriorate both the primal and the dual cost, although in the end they find an optimal primal solution. Their origin lies in nondifferentiable optimization (where nonmonotonic subgradient-based algorithms are common), and in the ϵ-subgradient method of [BeM73] (where the progress of the method depends on gradually reducing an ϵ parameter to an acceptable tolerance level). Auction algorithms are also highly intuitive and easy to understand; they can be explained in terms of economic competition concepts, which are couched on everyday experience.

Auction algorithms originated in 1979 with an algorithm proposed by the author for solving the classical assignment problem [Ber79]. The motivation was to solve the problem by using parallelism in a natural way. It turned out, however, that the resulting method was very fast in a serial environment as well. Subsequent work extended the auction algorithm to other linear network flow problems. The purpose of this paper is to provide a tutorial introduction to auction algorithms. We will initially focus on the basic algorithm for the assignment problem, and then discuss various extensions to other problems.

In the classical assignment problem there are n persons and n objects that we have to match on a one-to-one basis. There is a benefit a_{ij} for matching person i with object j and we want to assign persons to objects so as to maximize the total benefit. Mathematically, we want to find a one-to-one assignment (a set of person-object pairs $(1, j_1), \ldots, (n, j_n)$, such that the objects $j_1, \ldots, j_n$ are all distinct) that maximizes the total benefit $\sum_{i=1}^{n} a_{ij_i}$.

Consider the possibility of matching the n objects with the n persons through a market mechanism, viewing each person as an economic agent acting in his/her own best interest. Suppose that object j has a price p_j and that the person who receives the object must pay the price p_j. Then, the (net) value of object j for person i is $a_{ij} - p_j$ and each person i would logically want to be assigned to an object j_i with maximal value, that is, with

$$a_{ij_i} - p_{j_i} = \max_{j=1,\ldots,n} \{a_{ij} - p_j\}. \tag{1}$$

We say that an assignment and a set of prices satisfy *complementary slackness (or CS for short)* when this condition holds for all persons i. The economic system would then be at equilibrium, in the sense that no person would have an incentive to act unilaterally, seeking another object.

A basic linear programming result is that if a feasible assignment and a set of prices satisfy the complementary slackness conditions (1) for all persons i, then the assignment is optimal and the prices are an optimal solution of the following problem

$$\min_{\substack{p_j \\ j=1,\ldots,n}} \left\{ \sum_{j=1}^{n} p_j + \sum_{i=1}^{n} \max_{j} \{a_{ij} - p_j\} \right\}, \tag{2}$$

called the *dual problem*. Furthermore, the value of the optimal assignment and the optimal cost of the dual problem are equal.

2. THE NAIVE AUCTION ALGORITHM

Let us consider a natural process for finding an equilibrium assignment and price vector. We will call this process the *naive* auction algorithm, because it has a serious flaw, as will be seen shortly. Nonetheless, this flaw will help motivate a more sophisticated and correct algorithm.

The naive auction algorithm proceeds in iterations and generates a sequence of price vectors and partial assignments. By a *partial assignment* we mean an assignment where only a subset of the persons have been matched with objects. At the beginning of each iteration, the CS condition

$$a_{ij_i} - p_{j_i} = \max_{j=1,\ldots,n} \{a_{ij} - p_j\}$$

is satisfied for all pairs (i, j_i) of the partial assignment.

If all persons are assigned, the algorithm terminates. Otherwise a nonempty subset I of persons i that are unassigned is selected. Each of these persons, call him/her i, finds an object j_i which offers maximal value, that is,

$$j_i \in \arg \max_{j=1,\ldots,n} \{a_{ij} - p_j\}, \tag{3}$$

and computes a bidding increment

$$\gamma_i = v_i - w_i, \tag{4}$$

where v_i is the best object value,

$$v_i = \max_{j} \{a_{ij} - p_j\}, \tag{5}$$

and w_i is the second best object value

$$w_i = \max_{j \neq j_i} \{a_{ij} - p_j\}. \tag{6}$$

Each object j that is selected as best object by a nonempty subset $P(j)$ of persons in I, determines the highest bidder

$$i_j = \max_{i \in P(j)} \gamma_i, \tag{7}$$

raises its prices by the highest bidding increment $\max_{i \in P(j)} \gamma_i$, and gets assigned to the highest bidder i_j; the person that was assigned to j at the beginning of the iteration (if any) becomes unassigned.

This process is repeated in a sequence of iterations until all persons have an assigned object.

Note that γ_i cannot be negative since $v_i \geq w_i$ [compare Eqs. (5) and (6)], so the object prices tend to increase. Just as in a real auction, bidding increments and price increases spur competition by making the bidder's own preferred object less attractive to other potential bidders.

Note also that there is some freedom in choosing the subset of persons I that bid during an iteration. One possibility is to let I consist of a single unassigned person. This version, known as the *Gauss-Seidel version* because of its similarity with Gauss-Seidel methods for solving systems of nonlinear equations, usually works best in a serial computing environment. The version where I consists of all unassigned persons, is the one best suited for parallel computation; it is known as the *Jacobi version* because of its similarity with Jacobi methods for solving systems of nonlinear equations.

3. ϵ-COMPLEMENTARY SLACKNESS

Unfortunately, the naive auction algorithm does not always work (although it is an excellent initialization procedure for other methods, e.g. primal-dual or relaxation). The difficulty is that the bidding increment γ_i is zero when more than one object offers maximum value for the bidder i. As a result, a situation may be created where several persons contest a smaller number of equally desirable objects without raising their prices, thereby creating a never ending cycle; see Fig. 1.

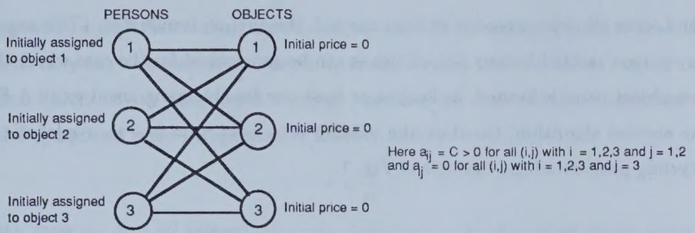
To break such cycles, we introduce a perturbation mechanism, motivated by real auctions where each bid for an object must raise the object's price by a minimum positive increment, and bidders must on occasion take risks to win their preferred objects. In particular, let us fix a positive scalar ϵ and say that a partial assignment and a price vector p satisfy *ϵ-complementary slackness (or ϵ-CS for short)* if

$$a_{ij_i} - p_{j_i} \geq \max_{j=1,\ldots,n} \{a_{ij} - p_j\} - \epsilon. \tag{8}$$

for all assigned pairs (i, j_i). In words, to satisfy ϵ-CS, all assigned persons of the partial assignment must be assigned to objects that are within ϵ of being best.

4. THE AUCTION ALGORITHM

We now reformulate the previous auction process so that the bidding increment is always at least


PERSONS
OBJECTS
Initially assigned to object 1
Initially assigned to object 2
Initially assigned to object 3
1
2
3
1
2
3
Initial price = 0
Initial price = 0
Initial price = 0
Here a_{ij} = C > 0 for all (i,j) with i = 1,2,3 and j = 1,2
and a_{ij} = 0 for all (i,j) with i = 1,2,3 and j = 3

At Start of Iteration #	Object Prices	Assigned Pairs	Bidder	Preferred Object	Bidding Increment
1	0,0,0	(1,1), (2,2)	3	2	0
2	0,0,0	(1,1), (3,2)	2	2	0
3	0,0,0	(1,1), (2,2)	3	2	0

Figure 1: Illustration of how the naive auction algorithm may never terminate for a three person and three object problem. Here objects 1 and 2 offer benefit $C > 0$ to all persons, and object 3 offers benefit 0 to all persons. The algorithm cycles as persons 2 and 3 alternately bid for object 2 without changing its price because they prefer equally object 1 and object 2 ($\gamma_i = 0$).

equal to ϵ. The resulting method, the *auction algorithm*, is the same as the naive auction algorithm, except that the bidding increment γ_i is

$$\gamma_i = v_i - w_i + \epsilon, \tag{9}$$

[rather than $\gamma_i = v_i - w_i$ as in Eq. (4)]. With this choice, the ϵ-CS condition is satisfied. The particular increment $\gamma_i = v_i - w_i + \epsilon$ used in the auction algorithm is the maximum amount with this property. Smaller increments γ_i would also work as long as $\gamma_i \geq \epsilon$, but using the largest possible increment accelerates the algorithm. This is consistent with experience from real auctions, which tend to terminate faster when the bidding is aggressive.

It can be shown that this reformulated auction process terminates in a finite number of iterations, necessarily with a feasible assignment and a set of prices that satisfy ϵ-CS. To get a sense of this, note that if an object receives a bid in m iterations, its price must exceed its initial price by at least $m\epsilon$. Thus, for sufficiently large m, the object will become "expensive" enough to be judged "inferior" to some object that has not received a bid so far. It follows that only for a limited number of iterations can an object receive a bid while some other object still has not yet received any bid.

On the other hand, once all objects receive at least one bid, the auction terminates. (This argument assumes that any person can bid for any object, but it can be generalized for the case where the set of feasible person-object pairs is limited, as long as at least one feasible assignment exists.) Figure 2 shows how the auction algorithm, based on the bidding increment $\gamma_i = v_i - w_i + \epsilon$ [cf. Eq. (9)], overcomes the cycling problem of the example of Fig. 1.

At Start of Iteration #	Object Prices	Assigned Pairs	Bidder	Preferred Object	Bidding Increment
1	0,0,0	(1,1), (2,2)	3	2	ϵ
2	0,ϵ,0	(1,1), (3,2)	2	1	2ϵ
3	2ϵ,ϵ,0	(2,3), (3,1)	1	2	2ϵ
4	2ϵ,3ϵ,0	(1,2), (2,1)	3	1	2ϵ
5	4ϵ,3ϵ,0	(1,3), (3,2)	2	2	2ϵ
6	...	...	...	...	...

Figure 2 Illustration of how the auction algorithm overcomes the cycling problem for the example of Fig. 1, by making the bidding increment at least equal to ϵ. The table shows one possible sequence of bids and assignments generated by the auction algorithm, starting with all prices equal to 0. At each iteration except the last, the person assigned to object 3 bids for either object 1 or 2, increasing its price by ϵ in the first iteration and by 2ϵ in each subsequent iteration. In the last iteration, after the prices of 1 and 2 rise at or above C, object 3 receives a bid and the auction terminates.

When the auction algorithm terminates, we have an assignment satisfying ϵ-CS, but is this assignment optimal? The answer here depends strongly on the size of ϵ. In a real auction, a prudent bidder would not place an excessively high bid for fear that he/she might win the object at an unnecessarily high price. Consistent with this intuition, we will show that if ϵ is small, then the final assignment will be "almost optimal." In particular, the following proposition (proved in several sources, e.g. [Ber79], [Ber88], [BeT89], [Ber91]) shows that *the total cost of the final assignment is within $n\epsilon$ of being optimal.* The idea is that a feasible assignment and a set of prices satisfying ϵ-CS also satisfy CS (and are therefore primal and dual optimal, respectively) for a *slightly perturbed* problem where all costs a_{ij} are the same as before, except for the costs of the n assigned pairs which are modified by an amount no more than ϵ.

Proposition 1: A feasible assignment satisfying ϵ-complementary slackness together with some price vector is within $n\epsilon$ of being optimal.

Suppose now that the costs a_{ij} are all integer, which is the typical practical case (if a_{ij} are rational numbers, they can be scaled up to integer by multiplication with a suitable common number). Then, the total benefit of any assignment is integer, so if $n\epsilon < 1$, any complete assignment that is within $n\epsilon$ of being optimal must be optimal. It follows, that *if*

$$\epsilon < \frac{1}{n},$$

and the costs a_{ij} are all integer, then the assignment obtained upon termination of the auction algorithm is optimal.

Figure 3 shows the sequence of generated object prices for the example of Fig. 1 in relation to the contours of the dual cost function. It can be seen from this figure that with each bid, the dual cost is approximately minimized (within ϵ) with respect to the price of the object receiving the bid. Successive minimization of a cost function along single coordinates is a central feature of coordinate descent and relaxation methods, which are popular for unconstrained minimization of smooth functions and for solving systems of smooth equations. Thus, the auction algorithm can be interpreted as an approximate coordinate descent method and as such, it is related to the relaxation method discussed in the previous subsection.

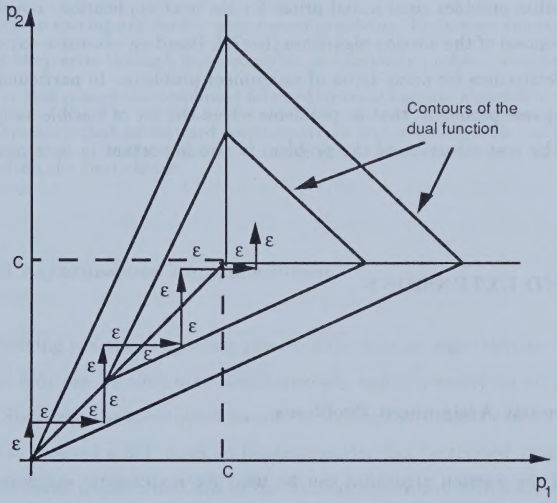


Figure 3: A sequence of prices p_1 and p_2 generated by the auction algorithm for the example of Fig. 1. The figure shows the equal dual cost surfaces in the space of p_1 and p_2, with p_3 fixed at 0.

5. COMPUTATIONAL ASPECTS – ϵ-SCALING

We note that the amount of work needed to solve the problem can depend strongly on the value of ϵ and on the maximum absolute object value

$$C = \max_{ij} |a_{ij}|.$$

Basically, for many types of problems, the number of iterations up to termination tends to be proportional to C/ϵ. This can be seen from the example of Fig. 3, where the number of iterations up to termination is roughly C/ϵ, starting from zero initial prices.

Note also that there is a dependence on the initial prices; if these prices are "near optimal," we expect that the number of iterations to solve the problem will be relatively small. This can be seen from the example of Fig. 3; if the initial prices satisfy $p_1 \approx p_3 + C$ and $p_2 \approx p_3 + C$, the number of iterations up to termination is quite small.

The preceding observations suggest the idea of *ϵ-scaling*, which consists of applying the algorithm several times, starting with a large value of ϵ and successively reducing ϵ up to an ultimate value that is less than some critical value (for example, $1/n$, when the benefits a_{ij} are integer). Each application of the algorithm provides good initial prices for the next application. ϵ-scaling was suggested in the original proposal of the auction algorithm [Ber79], based on extensive experimentation, which established its effectiveness for many types of assignment problems. In particular, ϵ-scaling is typically beneficial for sparse problems, that is, problems where the set of feasible assignment pairs is severely restricted. The cost structure of the problem is also important in determining whether ϵ-scaling is needed.

6. VARIATIONS AND EXTENSIONS

Extension to Asymmetric Assignment Problems

A slight variation of the auction algorithm can be used for asymmetric assignment problems where the number of objects is larger than the number of persons and there is a requirement that each person be assigned to some object. To solve this problem, the auction algorithm need only be modified in the choice of initial conditions. It is sufficient to require that all initial prices be zero. A similar algorithm can be used for the case where there is no requirement that all persons be assigned.

Reverse Auction

Unfortunately, the variation just described for the asymmetric assignment problem cannot be embedded in a scaling procedure because of the zero initial price restriction; to do ϵ-scaling, we must use the nonzero final prices obtained for a large value of ϵ to initialize the algorithm for a smaller value of ϵ. As a result the method performs poorly for problems where price wars (protracted sequences of small price changes of the type illustrated in Fig. 3) are likely to arise; such problems include sparse nearly symmetric assignment problems with moderately or very large cost range.

An interesting approach that circumvents this difficulty involves the idea of *reverse auction*, which was recently proposed in [BCT91]. While in the auction algorithm, persons compete for objects by bidding and raising the price of their best object, in reverse auction the objects compete for persons by essentially offering discounts (lowering their prices). One can show that forward and reverse auctions are mathematically equivalent, but their combination results in algorithms that can solve the symmetric and asymmetric assignment problems, as well as a variety of inequality-constrained assignment-like problems, more efficiently than forward auction alone.

The combination of forward and reverse auction is also far less susceptible to price wars than forward auction in solving symmetric assignment problems. Price wars can still occur in the combined algorithm, by they arise through more complex and unlikely problem structures than in the forward algorithm. For this reason the combined forward/reverse auction algorithm depends less on ϵ-scaling for good performance than its forward counterpart; in fact, starting with $\epsilon = 1/(n+1)$, thus bypassing ϵ-scaling, is often the best choice.

Parallel and Asynchronous Implementation

Both the bidding and the assignment phases of the auction algorithm are highly parallelizable. In particular the bids can be computed simultaneously and in parallel for all persons participating in the auction. Similarly, the subsequent awards to the highest bidders can be computed in parallel by all objects that received a bid. Such an implementation can be termed *synchronous*. There are also *totally asynchronous* implementations of the auction algorithm, which are interesting because they are quite flexible and also tend to result in faster solution in some types of parallel machines. In such implementations, a person may be viewed as an autonomous decision maker who at unpredictable times obtains information about the prices of the objects. Each unassigned person bids at an arbitrary time on the basis of his/her current object price information (that may be outdated because of communication delays).

A careful formulation of the totally asynchronous model, and a proof of its validity is given in [BeC89c], which includes also extensive computational results on a shared memory machine, confirming the advantage of asynchronous over synchronous implementations. Other parallel implementations of the auction algorithm have been presented in several studies [PhZ88], [KKZ89], [WeZ90], [Zak90].

Extension to Transportation and Minimum Cost Flow Problems

The auction algorithm can be extended to solve a variety of linear network flow problems. This is not surprising since the general linear minimum cost network flow problem can be transformed to an assignment problem (see e.g. [Ber91a], p. 17). By applying the auction algorithm to this problem and by streamlining the computations (sometimes in creative ways) one can obtain auction algorithms for network flow problems such as max-flow, shortest path, transportation, min-cost flow, etc.

In particular, the auction algorithm has been extended to solve linear transportation problems [BeC89a]. The basic idea is to convert the transportation problem into an assignment problem by creating multiple copies of persons (or objects) for each source (or sink respectively), and then to modify the auction algorithm to take advantage of the presence of the multiple copies. Computational results show that this auction algorithm is considerably faster than its chief competitors for important classes of transportation problems. Generally these problems are characterized by two properties, ***homogeneity*** and ***asymmetry***. A homogeneous problem is one for which there are only few levels of supply and demand. An asymmetric problem is one for which the number of sources is much larger than the number of sinks. For other types of transportation problems, the auction algorithm is outperformed by state-of-the-art codes based on relaxation methods; see [BeC89a], [Ber91].

There are extensions of the auction algorithm for linear minimum cost flow (transshipment) problems. The first such extension is the *ϵ-relaxation method* due to [Ber86a], [Ber86b]; see [BeT89] and [Ber91a] for a detailed description. The computational complexity of the scaled version of this algorithm was first studied in [Gol87], where particularly favorable polynomial running time estimates were derived; see [BeE87], [BeE88], [GoT90] for additional results along these lines. Serial implementations of the ϵ-relaxation method for general minimum cosst flow problems are not yet competitive with implementations of other methods such as relaxation; see [Ber91]. However, the ϵ-relaxation method is well suited for parallel computation. In fact the method admits a totally asynchronous implementation, as first shown in [Ber86a]; see also [BeE88], [BeT89]. A synchronous massively parallel implementation of the ϵ-relaxation method is presented in [LiZ91].

A general algorithm of the auction type for linear minimum cost flow problems was also given in [BeC89b]; it includes as special cases the auction algorithm for assignment and transportation problems, as well as the ϵ-relaxation method. By specializing this general algorithm to network flow problems with special structure, one may obtain efficient methods; for example, an interesting and intuitive algorithm for the k node-disjoint shortest path problem is developed in [BeC91].

Application to Max-Flow Problems

The ϵ-relaxation method can be applied to the max-flow problem, once this problem is converted to the minimum cost flow format, involving a feedback arc connecting the sink with the source, and having cost -1 (see [BeT89, [Ber91]). When this is done, the ϵ-relaxation method bears a close resemblance with a max-flow algorithm first proposed in [Gol85] and [GoT86]. These max-flow algorithms were derived from a different point of view that is unrelated to duality or ϵ-CS. They use node "labels," which in the context of the ϵ-relaxation approach can be viewed as prices. The max-flow version of the ϵ-relaxation method, first given in [Ber86a], [Ber86b] is simpler than the algorithms of [Gol85] and [GoT86] in that it obtains a maximum flow in one phase rather than two. It can also be initialized with arbitrary prices, whereas in the max-flow algorithms of [Gol85], [GoT86] the initial prices must satisfy $p_i \leq p_j + 1$ for all arcs (i, j). This can be significant, for example if one wants to use scaling or more generally, in a reoptimization setting where one wants to capitalize on the results of the solution of a slighly different max-flow problem.

In practice, the ϵ-relaxation method often finds a minimum cut very quickly – much faster than classical methods such as the Ford-Fulkerson algorithm. However, the method finds a maximum flow much faster than the Ford-Fulkerson algorithm for fairly dense problems and much slower for very sparse problems. This is due to the price war phenomenon referred to earlier. If one is interested in just a minimum cut or just the value of the maximum flow, it is worth testing periodically to see whether a minimum cut has already been obtained. A method for doing this is described in [Ber91] (Exercise 5.4) and is used in a max-flow code given in Appendix A.6 of [Ber91].

Extension to Shortest Path Problems

A new and simple auction algorithm for finding shortest paths in a directed graph was recently developed in [Ber91a], [Ber91b]. In the single origin/single destination case, the algorithm maintains a single path starting at the origin, which is either extended or contracted by a single node at each iteration. Simultaneously, at most one price variable is adjusted at each iteration so as to either improve or maintain the value of a dual function. The algorithm may be viewed as a special case of the naive auction algorithm applied to a special type of assignment problem, which is derived from the shortest path problem. While, as discussed earlier, the naive auction algorithm may not be successful in solving general assignment problems, it is guaranteed to solve this particular type of assignment problem thanks to its special structure.

Based on experiments with randomly generated problems on a serial machine, the auction/shortest path algorithm outperforms substantially its closest competitors for problems with few origins and a single destination (the computation time is reduced often by an order of magnitude or more). Also, for the case of multiple origins, the algorithm is better suited for parallel computation than other shortest path algorithms. The thesis [Pol91] discusses parallel asynchronous implementations of the algorithm.

7. CONCLUSIONS

Much progress has been made in the last few years to extend and apply auction algorithms to a variety of linear network flow problems, and to place them on an equal footing with the classical primal and dual cost improvement methods. At present, it appears that for simple problems such as assignment, shortest path, and max-flow, auction algorithms are at least competitive and often far superior to the classical methods; they are also better suited for parallel computation. Auction algorithms are also new and not fully developed. With further research, they should become more broadly applicable and more competitive with the classical methods.

REFERENCES

[BCT91] Bertsekas, D. P., Castañon, D. A., and Tsaknakis, H., 1991. "Reverse Auction and the Solution of Inequality Constrained Assignment Problems," Unpublished Report.

[BeC89a] Bertsekas, D. P., and Castañon, D. A., "The Auction Algorithm for Transportation Problems," Annals of Operations Research, Vol. 20, pp. 67-96.

[BeC89b] Bertsekas, D. P., and Castañon, D. A., "The Auction Algorithm for the Minimum Cost Network Flow Problem," Laboratory for Information and Decision Systems Report LIDS-P-1925, M.I.T., Cambridge, MA, (November).

[BeC89c] Bertsekas, D. P., and Castañon, D. A., "Parallel Synchronous and Asynchronous Implementations of the Auction Algorithm," Alphatech Report, Burlington, MA, (November), to appear in Parallel Computing.

[BeC90] Bertsekas, D. P., and Castañon, D. A., "Parallel Asynchronous Hungarian Methods for the Assignment Problem," Alphatech Report, Burlington, MA, Jan. 1990.

[BeC91] Bertsekas, D. P., and Castañon, D. A., "A Generic Auction Algorithm for the Minimum Cost Network Flow Problem," Alphatech Report, Burlington, MA, Sept. 1991.

[BeE87] Bertsekas, D. P., and Eckstein, J., "Distributed Asynchronous Relaxation Methods for Linear Network Flow Problems," Proc. of IFAC '87, Munich, Germany, July 1987.

[BeE88] Bertsekas, D. P., and Eckstein, J., "Dual Coordinate Step Methods for Linear Network Flow Problems," Math. Progr., Series B, Vol. 42, 1988, pp. 203-243.

[BeM73] Bertsekas, D. P., and Mitter, S. K., "Descent Numerical Methods for Optimization Problems with Nondifferentiable Cost Functions," SIAM Journal on Control, Vol. 11, pp. 637-652.

[BeT89] Bertsekas, D. P., and Tsitsiklis, J. N., Parallel and Distributed Computation: Numerical Methods, Prentice-Hall, Englewood Cliffs, N. J., 1989.

[Ber79] Bertsekas, D. P., "A Distributed Algorithm for the Assignment Problem," Lab. for Information and Decision Systems Working Paper, M.I.T., (March).

[Ber85] Bertsekas, D. P., "A Distributed Asynchronous Relaxation Algorithm for the Assignment Problem," Proc. 24th IEEE Conf. Dec. & Contr., pp. 1703-1704.

[Ber86a] Bertsekas, D. P., "Distributed Asynchronous Relaxation Methods for Linear Network Flow Problems," LIDS Report P-1606, M.I.T., (November).

[Ber86b] Bertsekas, D. P., "Distributed Relaxation Methods for Linear Network Flow Problems," Proceedings of 25th IEEE Conference on Decision and Control, pp. 2101-2106.

[Ber88] Bertsekas, D. P., "The Auction Algorithm: A Distributed Relaxation Method for the Assignment Problem," Annals of Operations Research, Vol. 14, pp. 105-123.

[Ber91a] Bertsekas, D. P., Linear Network Optimization: Algorithms and Codes, M.I.T. Press, Cambridge, Mass.

[Ber91b] Bertsekas, D. P., "An Auction Algorithm for Shortest Paths," SIAM J. on Optimization, Vol. 1, (October).

[Gol85] Goldberg, A. V., "A New Max-Flow Algorithm," Tech. Mem. MIT/LCS/TM-291, Laboratory for Computer Science, M.I.T., Cambridge, MA.

[GoT86] Goldberg, A. V., and Tarjan, R. E., "A New Approach to the Maximum Flow Problem," Proc. 18th ACM STOC, pp. 136-146.

[GoT90] Goldberg, A. V., and Tarjan, R. E., "Solving Minimum Cost Flow Problems by Successive Approximation," Math. of Operations Research, Vol. 15, pp. 430-466.

[Gol87] Goldberg, A. V., "Efficient Graph Algorithms for Sequential and Parallel Computers," Tech. Report TR-374, Laboratory for Computer Science, M.I.T., Cambridge, MA.

[KKZ89] Kempa, D., Kennington, J., and Zaki, H., "Performance Characteristics of the Jacobi and Gauss-Seidel Versions of the Auction Algorithm on the Alliant FX/8," Report OR-89-008, Dept. of Mech. and Ind. Eng., Univ. of Illinois, Champaign-Urbana, 1989.

[LiZ91] Li, X., and Zenios, S. A., "Data Parallel Solutions of Min-Cost Network Flow Problems Using ϵ-Relaxations," Report 1991-05-20, Dept. of Decision Sciences, The Wharton School, Univ. of Pennsylvania, Phil., Penn.

[PhZ88] Phillips, C., and Zenios, S. A., "Experiences with Large Scale Network Optimization on the Connection Machine," Report 88-11-05, Dept. of Decision Sciences, The Wharton School, Univ. of Pennsylvania, Phil., Penn., Nov. 1988.

[Pol91] Polymenakos, L., "Analysis of Parallel Asynchronous Schemes for the Auction Shortest Path Algorithm," MS Thesis, EECS Dept., M.I.T., Cambridge, MA.

[Roc84] Rockafellar, R. T., Network Flows and Monotropic Programming, Wiley-Interscience, N. Y.

[WeZ90] Wein, J., and Zenios, S. A., "Massively Parallel Auction Algorithms for the Assignment Problem," Proc. of 3rd Symposium on the Frontiers of Massively Parallel Computation, Md.

[Zak90] Zaki, H., "A Comparison of Two Algorithms for the Assignment Problem," Report ORL 90-002, Dept. of Mechanical and Industrial Engineering, Univ. of Illinois, Urbana, Ill.

Research supported by NSF under Grant No. DDM-8903385 and by the ARO under Grant DAAL03-86-K-0171.

Generalized Intersection Searching Problems

Ravi Janardan[*]
Mario Lopez[†]

Department of Computer Science
University of Minnesota
Minneapolis, MN 55455

September 18, 1991

Abstract

A new class of geometric intersection searching problems is introduced, which generalizes previously-considered intersection searching problems and is rich in applications. In a standard intersection searching problem, a set S of n geometric objects is to be preprocessed so that the objects that are intersected by a query object q can be reported efficiently. In a generalized problem, the objects in S come aggregated in disjoint groups and what is of interest are the groups, not the objects, that are intersected by q. Although this problem can be solved easily by using an algorithm for the standard problem, the query time can be $\Omega(n)$ even though the output size is just $O(1)$. In this paper, space-efficient algorithms with low, output-size-sensitive query times are presented for the generalized versions of several intersection searching problems, including: interval intersection searching, orthogonal segment intersection searching, point enclosure searching, and segment intersection searching.

1 Introduction

1.1 The problem and its significance

Intersection searching problems are among the most widely studied classes of problems in computational geometry [1, 3, 5, 6, 10, 12, 15]. In a standard intersection searching problem, a set, S, of n geometric objects is to be preprocessed so that the objects that are intersected by a query object q can be reported efficiently. Examples of (S, q)-pairs include: (intervals, interval) — *interval intersection searching*, (line segments, line segment) — *segment intersection searching*, (axes-parallel rectangles, points) — *point enclosure searching*, and (points, axes-parallel rectangle) — *orthogonal range searching*.

---

[*] Research supported in part by a grant-in-aid of research, by a Faculty Summer Research Fellowship, by the Army High Performance Computing Research Center (all at the University of Minnesota), and by NSF grant CCR-8808574. E-mail: janardan@cs.umn.edu; Phone: (612)-625-7338

[†] Research supported in part by grants DCR 84-20935 and MIP 86-17374. E-mail: lopez@cs.umn.edu; Phone: (612)-625-0519

In many applications, a more general problem arises, where the objects of S come aggregated in disjoint groups and what is of interest are the groups, not the objects, intersected by q. (q intersects a group iff it intersects some object in the group.) For lack of space, we give just two examples: [VLSI LAYOUT:] In the (Manhattan) layout of a VLSI chip, the wires can be grouped naturally into circuits. Determining the circuits that become electrically connected when a new wire is added is precisely the generalized (orthogonal) segment intersection searching problem. [DATABASE QUERYING:] Consider the database of a phone company which contains the time of day and the duration of every phone call made by its customers over a period of time. Thus, calls can be grouped by customers. To determine usage patterns (say, to market calling plans), one would like to identify the customers making calls during specific times and for specific durations. This is just the generalized orthogonal range searching problem.

Intuitively, if we imagine that each object of S is assigned a color according to its group, then our problem is to report the distinct colors of the objects intersected by q. Thus, we will call S a *chromatic set* and each group a *color class*. Note that the generalized problem reduces to the standard one when each color class has cardinality 1.

Although a generalized problem can be solved easily by using a query algorithm for the underlying standard problem, the query time can be $\Omega(n)$ even though the output size, i, for the generalized problem, i.e., the number of color classes intersected by q, is only $O(1)$. The challenge in the generalized problem is to achieve a query time that is sensitive to the output size, while keeping the input-size component of the query time and the space small. Thus, in addition to its many applications, the problem is also interesting from the algorithm design standpoint.

1.2 Overview of results

In this paper, we present space-efficient algorithms with low, output-size-sensitive query times for a number of generalized intersection searching problems, as summarized in Table 1.

GENERALIZED PROBLEM	SPACE	QUERY TIME
Interval intersection searching	n	$\log n + i$
Orthogonal segment intersection searching	$n \log n$	$\log n + i$
	n	$\log^2 n + i$
Point enclosure searching in $\mathcal{R}^d$, $d \geq 2$ is a small constant **		
$d = 2$	$n^{1.5}$	$\log n + i$
$d = 3$	$n^{1.75}$	
...	...	
Segment intersection searching (non-intersecting line segments)	$n^2 \log n$	$\log n + i$
	n^2	$\log^2 n + i$
Orthogonal range searching in $\mathcal{R}^2$	$n \log^2 n$	$\log n + i$
	$n \log n$	$\log^2 n + i$
Segment intersection searching (intersecting segments; query line)	n^2	$\log n + i$
If color classes are connected segments	$n \log^2 n$	$\min\{\sqrt{n} \log^2 n + i, \sqrt{n} \log n + i \log n\}$
	$n \log n$	$n^\gamma \log n + i$, $\gamma \approx 0.695$
Rectangle intersection searching	$n\sqrt{n}$	$\log n + i$
If color classes correspond to connected components	$n \log n$	$\log n + i$
	$n \log n / \log \log n$	$\log^2 n + i$

Table 1: *Summary of results for generalized intersection searching problems. All bounds given are "big-oh" and worst-case. Here i denotes the output size, i.e., the number of color classes intersected.*

** More precisely, the query time is $O(d!(\log n + i))$. Hence, the restriction that d be a small constant. (The algorithm itself works for any d.) For constant d, the space bound is $O(n^{\beta_d})$, where $\beta_2 = 1.5$ and, for $d > 2$, $\beta_d = \beta_{d-1} + \frac{1}{1+(d-1)\beta_{d-1}}$.

Very roughly, our strategy is to generate a suitable representation for each color class with the properties that (1) the representation is "sparse" so that q intersects it only $O(1)$ times and (2) q intersects the color class iff q intersects the representation. We then suitably organize the representations for all color classes into an intersection searching structure which we query with q. Property 2 guarantees that exactly the color classes intersected by q are reported, while property 1 ensures that only $O(1)$ intersections are found for each such color class, thus making the query time output-sensitive. (Multiple reports of a color class are avoided by using a bit-vector to track reported colors.)

Guaranteeing the "only if" part of Property 2 requires some care because q may be so "small" that it intersects a member of a color class but not the representation. We organize the representations hierarchically, so that at some level in the hierarchy q appears sufficiently "long". The only exception is point enclosure searching, where q is a point; here we resort to a different approach based on a partitioning of space.

For lack of space, we will discuss here only the first four problems in Table 1 and will omit all proofs and details. A full version of our paper is available as [8].

2 Generalized interval intersection searching

Theorem 1 *Let S be a chromatic set of n intervals on the real line (say, the x-axis). S can be preprocessed into a structure of size $O(n)$ such that the i distinct colors of the intervals of S that are intersected by a query interval, q, can be reported in $O(\log n + i)$ time. The preprocessing time is $O(n \log n)$. All bounds are worst-case.* □

Clearly, the problem can be solved by reporting the distinct colors of the intervals that contain one of q's endpoints and the distinct colors of the interval endpoints contained in q. These are instances of generalized 1-dimensional point enclosure and range searching, respectively. We show how each can be solved in $O(n)$ space and $O(\log n + i)$ query time.

Point enclosure searching is easy. We first replace overlapping intervals of the same color by their union, so that the intervals of each color become disjoint. All intervals (at most n) are stored in a window list. (A *window list* [1] for n intervals can be built in $O(n \log n)$ time, uses $O(n)$ space, and reports the k intervals intersected by a query point (or, a query interval) in $O(\log n + k)$ time.) Given a query point, we simply report the colors of the intersected segments. Since the point can intersect at most one interval of a color, the query time is $O(\log n + i)$.

Consider generalized range searching, where the input set, S, consists of points and q is an interval. We first replace like-colored points at the same x-coordinate by a single such point. We then set up a balanced binary search tree T with root r, as follows: We split S about its median x-coordinate, x_{med}, into sets S', S_l and S_r, where S' consists of the points at x_{med}, and S_l (resp. S_r) contains the points to the left (resp. right) of x_{med}. We store S' at r in a list, $M(r)$, along with the point $X(r) = x_{med}$. We generate a sparse representation for S_l (resp. S_r) by picking, for each color, the rightmost (resp. leftmost) point of that color in it and storing the points picked in a list $L(r)$ (resp. $R(r)$) in non-increasing (resp. non-decreasing) order. The left and right subtrees of T are set up recursively on S_l and S_r, respectively. Next, to attain the $O(n)$ space bound, we trim the lists $L(u)$ and $R(u)$ at each node u by retaining a point in $L(u)$ (resp. $R(u)$) iff the point has not been stored in $L(u')$ (resp. $R(u')$) for some ancestor u' of u. The points in $L(u)$ (and, similarly, $R(u)$) are stored at the leaves of a balanced binary search tree and the leaves are threaded into a doubly-linked list.

Given a query interval q, we do a binary search down T until either the search runs off T or a node v is reached such that q intersects $X(v)$. At each node u on the search path, if q is to the left (resp. right) of $X(u)$, then we locate x_1 in $L(u)$ (resp. $R(u)$) by binary search, and report the color of each point between x_1 and x_2 in the threaded list of leaves. If v exists then we output the colors of the points in $M(v)$ and then scan $L(v)$ and $R(v)$, reporting the colors of the points there intersected by q. It can be proved that a color c is reported iff q intersects a point of that color. and, moreover, that at most five points of any color are intersected during the search. Since there are $O(\log n)$ binary searches, the query time is $O(\log^2 n + i)$.

Figure 1 illustrates the data structure on a problem instance involving three colors. Each node is labeled by the unique X-value stored at it and each point in an L, M, or R list is identified by its coordinate and color. For the query interval $q = [0.5, 2.5]$, the search visits nodes 6 and 2. At 6, $L(6)$ is searched and the color G is output because q intersects 1G. At 2, color B is output from $M(2)$. The searches on $L(2)$ and $R(2)$ cause no new colors to be output.

Finally, we reduce the query time to $O(\log n + i)$, while retaining the $O(n)$ space bound, by applying the technique of *layering* as described in [1, page 718]. The entire structure can be built in $O(n \log n)$ time.

This proves Theorem 1. We remark that an $O(n)$-space and $O(\log n + i)$-query time solution to the problem was first given by M. Lopez and S. Sahni in a working paper [9]. The alternative solution given above was discovered subsequently by R. Janardan, independent of [9], and is presented here to make the present paper self-contained.

3 Generalized orthogonal segment intersection searching

Theorem 2 *Let S be a chromatic set of n orthogonal line segments in the plane. S can be preprocessed into a structure of size $O(n \log n)$ (resp. $O(n)$) such that the i distinct colors of the segments of S that are intersected by an orthogonal query segment q can be reported in $O(\log n + i)$ (resp. $O(\log^2 n + i)$) time. The preprocessing time is $O(n \log n)$. All bounds are worst-case.* □

Assume q is vertical. Let $H \subseteq S$ be the horizontal segments, preprocessed so that segments of the same color are disjoint. First, consider the special case where q is the semi-infinite line $[(x_0, y_1), (x_0, -\infty)]$. We generate the following sparse representation: Let $H_c \subseteq H$ be the segments of color c and let the *lower shell* of H_c be the subsegments visible from below H_c. (See Figure 2.) Each shell has at most $2|H_c| - 1$ segments, and any vertical line intersects at most two segments of the shell. We preprocess the lower shell segments of all colors into a hive graph. (A *hive graph* [1] on n horizontal segments can be built in $O(n \log n)$ time, uses $O(n)$ space, and can report the k segments intersected by a vertical query segment in $O(\log n + k)$ time.) We query the hive graph with q and report the color of each intersected segment. Since q is infinite downwards, it intersects a segment in H_c iff it intersects the lower shell of H_c. The query time is $O(\log n + i)$. An upward-infinite query segment is handled similarly (using upper shells, which are defined symmetrically).

To handle a finite query segment $q = [(x_0, y_1), (x_0, y_2)]$, $y_1 \geq y_2$, we store the distinct y-coordinates of the segments in H at the leaves of a balanced binary search tree T. For any node v, let $y(v)$ be the average of the largest y-coordinate in v's left subtree and the smallest y-coordinate in v's right subtree. (For a leaf v, $y(v)$ is just the associated y-coordinate.) Let $L(v)$ be the line $y = y(v)$. At a leaf v, we store its segments in a window list. For a non-leaf v, we store the segments in its right (resp. left) subtree in the structure described above for a downward-infinite (resp. upward-infinite) query. Call this structure $D_1(v)$ (resp. $D_2(v)$). The total space is $O(n \log n)$ because a shell segment can be stored at $O(\log n)$ levels.

Given q, we binary search down T until the search either runs off T or we find a node v such that q intersects $L(v)$. If v exists and is a leaf, then we query its window list with the point $(x_0, y(v))$ and report the color of each intersected segment. If v is a non-leaf, we query both $D_1(v)$ and $D_2(v)$ with q. Correctness follows because if q intersects $L(v)$, it is semi-infinite w.r.t. the region of the plane above and below $L(v)$. The query time is $O(\log n + i)$.

We can reduce the space to $O(n)$ by processing the structure top-down and storing at any level only those pieces of the lower (resp. upper) shell segments generated at that level that are disjoint (except for endpoints) from pieces of lower (resp. upper) shell segments stored at higher levels. Because shell pieces are disjoint, we can use a counting argument (see [8]) to upper-bound the total number of lower and upper shell pieces by $O(n)$. $D_1(v)$ (resp. $D_2(v)$) are now built on the lower (resp. upper) shell pieces generated at v. The query algorithm is as before, except that at each node u preceding v on the search path, if q is above (resp. below) $L(u)$, then we query $D_1(u)$ (resp. $D_2(u)$) with q. The query time is $O(\log^2 n + i)$.

Both schemes can be built easily in $O(n \log^2 n)$ time. This can be reduced further to $O(n \log n)$ by carefully combining hive graph computations at different levels (see [8]). Reporting intersections of q with vertical segments of S just boils down to the algorithm in Section 2.

4 Generalized point enclosure searching in $\mathcal{R}^d$

Theorem 3 *Let S be a chromatic set of n hyperrectangles in d-dimensional space, where $d \geq 2$ is a small constant. S can be preprocessed into a data structure of size $O(n^{\beta_d})$, such that the i distinct colors of the hyperrectangles that contain a query point q can be reported in $O(\log n + i)$ time (more precisely, $O(d!(\log n + i))$ time). Here $\beta_2 = 1.5$ and $\beta_d = \beta_{d-1} + \frac{1}{1+(d-1)\beta_{d-1}}$ for $d \geq 2$. Thus, for instance, $\beta_3 = 1.75$, $\beta_4 \approx 1.91$, $\beta_5 \approx 2.03$ etc.* □

First consider $d = 2$. We use a plane-partitioning technique that is reminiscent of the one described in [13] for a different problem. We sort the $2n$ vertical sides of the rectangles by x-coordinate, breaking ties arbitrarily. We then group these sides into $2\sqrt{n}$ *slabs* of $\sqrt{n}$ sides each and then divide each slab into *cells* by drawing a horizontal line within the slab through each endpoint of a vertical side in the slab. It is easy to check that: (1) no cell contains endpoints in its interior; (2) there are $O(n)$ cells in all; and (3) for any cell, there are only $O(\sqrt{n})$ rectangles whose horizontal sides partially cover the x-span of the cell.

Let s be any slab and let b be any cell of s. Let $H_{s,b}$ be the set of rectangles of S whose horizontal sides partially cover b's x-span. Let V_s be the rectangles of S that cross s completely. Note that $|H_{s,b}|$ is $O(\sqrt{n})$ but $|V_s|$ can be $O(n)$. For the rectangles in

$H_{s,b}$, we compute the x-projections of their horizontal sides as restricted to b's x-span and organize these into a window list structure, $W_{s,b}$, for generalized 1-dimensional point enclosure searching (Section 2). We organize the y-projections of the rectangles in V_s into another such structure W_s. The total space is $O(n\sqrt{n})$.

To answer a query we locate by binary search the slab s that contains q and then query W_s using the y-projection of q. Next, we locate by binary search the cell b of s that contains q and query $W_{s,b}$ using q's x-projection. Note that (1) within s each rectangle of V_s is unbounded in x and so q is in a rectangle of V_s iff q's y-projection is in that rectangle's y-projection, and (2) within a cell b, each rectangle of $H_{s,b}$ is unbounded in y and so q is in a rectangle of $H_{s,b}$ iff q's x-projection is in that rectangle's x-projection restricted to b's x-span. This proves correctness. The query time is $O(\log n + i)$.

For $d \geq 3$, we reduce a d-dimensional problem to a number of $(d-1)$-dimensional subproblems. Let $x_1, x_2, \ldots, x_d$ be the coordinate directions. Let α_d, $0 < \alpha_d < 1$, be a real-valued parameter to be fixed later. Call a hyperrectangle face (i.e. a *hyperface*) x_i-*normal* if it is normal to the x_i-direction, $1 \leq i \leq d$. Partition the d-dimensional space along the x_1 direction into $2n^{\alpha_d}$ regions, called x_1-*regions*, each containing $n^{1-\alpha_d}$ x_1-normal hyperfaces. Next, partition each x_1-region along the x_2-direction into $2n^{\alpha_d}$ x_2-*regions*, each containing at most $n^{1-2\alpha_d}$ x_2-normal hyperfaces of hyperrectangles that have x_1-normal hyperfaces in the x_1-region. Similarly, for $j = 3, 4, \ldots, d-1$, partition each x_{j-1}-region along the x_j direction. Finally, partition each x_{d-1}-region, s, into x_d-*regions* by drawing (within s) a hyperplane through each x_d-normal hyperface of a hyperrectangle that has an x_{d-1}-normal hyperface in s. It can be verified that: (1) the x_d-regions contain no hyperrectangle vertices in their interior; (2) there are at most $2^d n$ x_d-regions in all; and (3) for any cell, there are at most $n^{1-(d-1)\alpha_d}$ hyperrectangles whose x_d-normal hyperfaces only partially cover the $(x_1, x_2, \ldots, x_{d-1})$-span of the cell.

Let q be in an x_1-region r_1. Consider the set, R_1, of hyperrectangles that do not have an x_1-normal hyperface in r_1. Since the hyperrectangles in R_1 are effectively unbounded in the x_1 direction, the distinctly-colored ones in R_1 that contain q can be found by projecting the hyperrectangles of R_1 and the point q onto the $(x_2, x_3, \ldots, x_d)$-plane and doing recursively a generalized $(d-1)$-dimensional point enclosure query. The set R_1' of hyperrectangles that do have an x_1-normal hyperface in r_1 is handled by first locating the x_2-region, r_2, of r_1 that contains q. Let R_2 be the hyperrectangles of R_1' that do not have an x_2-normal hyperface in r_2. Since the hyperrectangles of R_2 are unbounded in the x_2 direction, the distinctly-colored members of R_2 that contain q can be found by projecting all the members of R_2 and q onto the $(x_1, x_3, x_4, \ldots, x_d)$-plane and doing a generalized $(d-1)$-dimensional point enclosure query. We proceed this way until we get to the x_d-region containing q. Any hyperrectangles that could contain q but which have not become unbounded w.r.t. the first $d-1$ directions will become unbounded w.r.t. the x_d direction (by property (1) above of an x_d-region) and will thus be found.

We recursively set up the $(d-1)$-dimensional data structures implied above. It can be shown that for d a small constant, the space is $O(n^{\beta_{d-1}+\alpha_d} + n^{1+(1-(d-1)\alpha_d)\beta_{d-1}})$. Choosing $\alpha_d = \frac{1}{1+(d-1)\beta_{d-1}}$ makes the two terms equal and gives a space bound of $O(n^{\beta_d})$, where $\beta_d = \beta_{d-1} + \alpha_d$. The query time sums up to $O(d!(\log n + i))$ since there are d queries on $(d-1)$-dimensional subproblems.

5 Generalized segment intersection searching

Theorem 4 *Let S be a chromatic set of n non-intersecting (but possibly touching) line* *segments in the plane. S can be preprocessed into a structure of size $O(n^2 \log n)$ (resp. $O(n^2)$) such that the i distinct colors of the segments that are intersected by a query segment can be reported in $O(\log n + i)$ (resp. $O(\log^2 n + i)$) time. The preprocessing time is $O(n^2 \log n)$. All bounds are worst-case.* □

We review briefly the solution in [1] for the corresponding standard problem. Let $q = \alpha\beta$ and let K be its supporting line. Let H be a line that is translated in a direction parallel to K until it encounters a segment endpoint, p. Thus the region enclosed by K and H contains no endpoints in its interior. The possibilities for p are shown in Figure 3. In Figure 3a, the segments intersected by q are precisely those that are intersected by the segments αp and βp, while in Figure 3b they are the segments that are intersected by a proper subsegment of βp that starts at β. To answer a query, the endpoint p is first determined and the appropriate case identified, all in $O(\log n)$ time with suitable preprocessing. Let G be a precomputed hive graph centered at p. In the case of Figure 3a, G is queried with βp and αp and all intersected segments are reported. In the case of Figure 3b, G is queried with βp, but the search is stopped as soon as a segment that is not intersected by q is found. The space is $O(n^2)$ and the query time is $O(\log n + k)$, where k is the output size.

Thus the problem of querying with an arbitrary segment is reduced to querying with two segments that are *centered w.r.t. p*, i.e., have their supporting lines passing through p. For the generalized problem, we will use a similar approach but in addition we need to find a way to organize the segments hierarchically and develop a suitable querying strategy. Moreover, segments of S centered w.r.t. p need to be handled separately. Let C be the set of such segments and let $N = S - C$. We first discuss how to handle N.

To organize the segments of N hierarchically, we use a total order, $\ll$, which was defined in [1] for constructing a centered hive graph. Let s_i and s_j be segments of N. Say that $s_i \preceq s_j$ if there is a ray emanating from p that hits both s_i and s_j, s_i before s_j. The relation $\preceq$ may contain cycles, but it can be made acyclic. Furthermore, it is possible to compute a subset $\preceq^\star$ of $\preceq$ with the property that it contains only $O(n)$ pairs and its transitive closure is the same as that of $\preceq$. The relation $\ll$ is obtained by doing a topological sort on $\preceq^\star$.

We also need the notion of shells. Let $N' \subseteq N$. Intuitively, the *inner shell of N' w.r.t. p* consists of the subsegments of N' visible 360° around p. The *outer shell of N' w.r.t. p* consists of the subsegments of N' visible when viewing N' from the "outside", i.e., when walking along the circumference of a circumscribing circle of N' centered at p.

Let $s_1, s_2, \ldots$ be the segments of N ordered according to $\ll$. We store $s_1, s_2, \ldots$ at the leaves of a balanced binary search tree, T, in left to right order. Let v be any internal node. For each color c, we compute for the segments in v's right (resp. left) subtree the inner (resp. outer) shell w.r.t. p. We then compute for the inner shell segments of all colors (and similarly for the outer shell segments) a hive graph centered at p and store these two hive graphs at v. We also store at v a value $I(v) = j + \frac{1}{2}$, where s_j is the segment stored at the rightmost leaf in v's left subtree.

Given $q = \alpha\beta$, consider first the case in Figure 3b. Let s_{i_1} and s_{i_t} be, respectively, the first and last segments of N intersected by q, when taken in order from α to β. Let γ (resp. δ) be the point of intersection between βp and s_{i_1} (resp. s_{i_t}). The segment $\gamma\delta$ intersects exactly the same segments as q and in the same order. Moreover, the indices of these segments are in $\{i_1, i_1 + 1, \ldots, i_t\}$. We can compute s_{i_1} and s_{i_t} (and hence γ and δ) in $O(\log n)$ time as follows. Let G be the hive graph centered at p and constructed on N.

Using G, we compute s_{i_t} as the first segment of N that is intersected by both βp, when βp is directed from β to p, and by q. Similarly, we can compute the first segment s' of N intersected by αp, when αp is directed from α to p. Let ϕ be the point of intersection between βp and s'. We can now compute s_{i_1} as the first segment of N that is intersected by $\phi\beta$, when $\phi\beta$ is directed from ϕ to β.

If s_{i_1} and s_{i_t} do not exist, then q intersects no segment of N. Otherwise, if $s_{i_1} = s_{i_t}$, then we only report the color of s_{i_1}. Otherwise, we do a binary search down T by comparing i_1 and i_t with the $I(\cdot)$ values of nodes encountered and stop as soon as a node v is reached such that $i_1 < I(v) < i_t$. We query the hive graphs built on the inner (and, similarly, the outer) shells with $\gamma\delta$ and report the color of each shell segment intersected. In the case of Figure 3a, we proceed as above, with both βp and αp. The query time is $O(\log n + i)$. The structure at p uses $O(n \log n)$ space. This can be reduced to $O(n)$ by storing disjoint pieces of shell segments at each level, as in the orthogonal case. However, the query time increases to $O(\log^2 n + i)$. Both structures can be set up in $O(n \log n)$ time.

We handle C as follows. Let C' consist of the segments of C that have p as an endpoint or pass through p. Let W be the sector of the plane that is bounded by αp and βp and contains q. The segments of C that intersect q are precisely the ones in C' that lie in W. This follows since the region between the lines K and H (Figure 3) contains no endpoints in its interior. We reduce our problem to generalized 1-dimensional range searching, as follows: In preprocessing, we associate with each segment of C' the angle that it makes at p, say, measured clockwise from a leftward-directed horizontal ray through p. The segments that pass through p are broken into two at p and treated as segments with different angles. Let a_α (resp. a_β) be the angle made by αp (resp. βp) at p. If $a_\alpha \leq a_\beta$, then we do a generalized 1-dimensional range query with the interval $[a_\alpha, a_\beta]$. Otherwise, we do two such queries, one with the $[a_\alpha, 360°]$ and the other with $[0°, a_\beta]$.

Thus the space used at p is dominated by the $O(n \log n)$ (resp. $O(n)$) space for N, and the total space is $O(n^2 \log n)$ (resp. $O(n^2)$).

6 Conclusions and open problems

The contributions of this paper have been (1) the introduction of a new class of intersection searching problems which generalize previously-considered intersection problems and are rich in applications and (2) the provision of a uniform framework for solving these problems which consists of generating a "sparse" representation of the input which captures all the essential information and then applying a standard intersection algorithm.

Two problems need to be investigated further. First, is it possible to solve a generalized intersection problem within the same space and query time bounds as the corresponding standard problem? We have shown this to be true only for interval intersection searching. Whether this is possible for any of the other problems is an interesting open question.

Second, the solutions given here cannot efficiently handle updates in the underlying set. Efficient dynamic solutions have been given for the standard versions of many of these problems (see [2, 3, 4, 7, 11, 15, 16] for a sampling of such results). Such solutions usually take advantage of the O(1)-time decomposability of the queries [11, 14]. The queries for the generalized problems are not O(1)-time decomposable; determining the answer to a problem from two subproblems requires eliminating duplicate colors, which can be expensive. Thus, new dynamization techniques need to be developed.

References

[1] B.M. Chazelle. Filtering search: a new approach to query-answering. *SIAM Journal on Computing*, 15:703–724, 1986.

[2] S.W. Cheng and R. Janardan. Efficient dynamic algorithms for some geometric intersection problems. *Information Processing Letters*, 36:251–258, 1990.

[3] S.W. Cheng and R. Janardan. Algorithms for ray-shooting and intersection searching. *Journal of Algorithms*, 1991. Accepted.

[4] H. Edelsbrunner. Dynamic data structures for orthogonal intersection queries. Technical Report 59, Technische Universität Graz, October 1980.

[5] H. Edelsbrunner, D. Kirkpatrick, and H.A. Maurer. Polygonal intersection searching. *Information Processing Letters*, 14:74–79, 1982.

[6] H. Edelsbrunner and H. Maurer. On the intersection of orthogonal objects. *Information Processing Letters*, 13:177–181, 1981.

[7] H. Imai and T. Asano. Dynamic orthogonal segment intersection search. *Journal of Algorithms*, 8:1–18, 1987.

[8] R. Janardan and M. Lopez. Generalized intersection searching problems. Technical Report TR–90–54 (revised), Dept. of Computer Science, University of Minnesota, 1991. Also submitted.

[9] M. Lopez and S. Sahni. Algorithms for static labeled intervals. Manuscript, 1990.

[10] E.M. McCreight. Priority search trees. *SIAM Journal on Computing*, 14:257–276, 1985.

[11] M.H. Overmars. *The design of dynamic data structures.* Springer–Verlag, 1983.

[12] M.H. Overmars, H. Schipper, and M. Sharir. Storing line segments in partition trees. Technical Report RUU–CS–89–17, University of Utrecht, August 1989.

[13] M.H. Overmars and C.K. Yap. New upper bounds in Klee's measure problem. In *Proceedings of the 29th Annual IEEE Symposium on Foundations of Computer Science*, pages 550–556, 1988.

[14] J.B. Saxe and J.L. Bentley. Transforming static data structures into dynamic data structures. In *Proceedings of the 20th Annual IEEE Symposium on Foundations of Computer Science*, pages 148–168, 1979.

[15] V.K. Vaishnavi and D. Wood. Rectilinear segment intersection, layered segment trees, and dynamization. *Journal of Algorithms*, 3:160–176, 1982.

[16] D.E. Willard and G.S. Lueker. Adding range restriction capability to dynamic data structures. *Journal of the ACM*, 32:597–617, 1985.

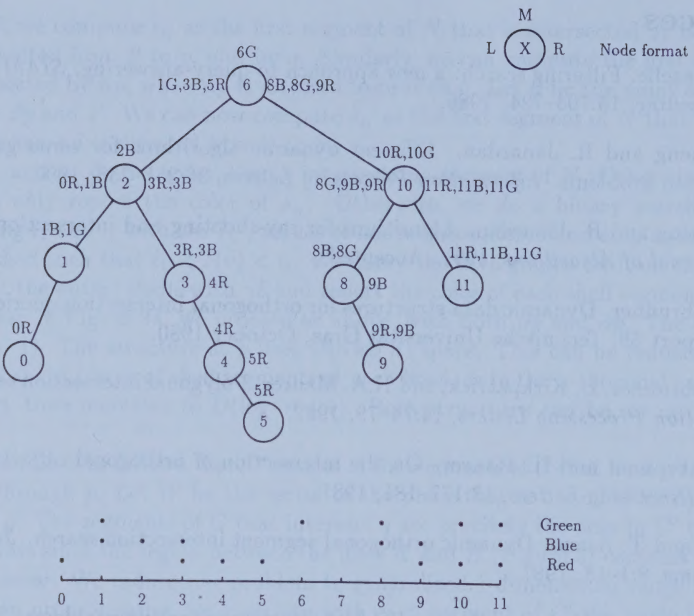

M
L
X
R
Node format
6G
1G,3B,5R
6
8B,8G,9R
2B
0R,1B
2
3R,3B
10R,10G
8G,9B,9R
10
11R,11B,11G
1B,1G
1
3R,3B
3
4R
8B,8G
8
9B
11R,11B,11G
11
0R
0
4R
4
5R
9R,9B
9
5R
5
Green
Blue
Red
0
1
2
3
4
5
6
7
8
9
10
11

Figure 1: Illustration of the data structure for generalized interval intersection searching.

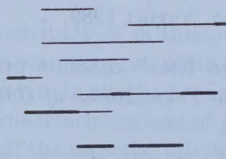


Figure 2: Illustration of the lower shell (shown bold) of a set of horizontal line segments.

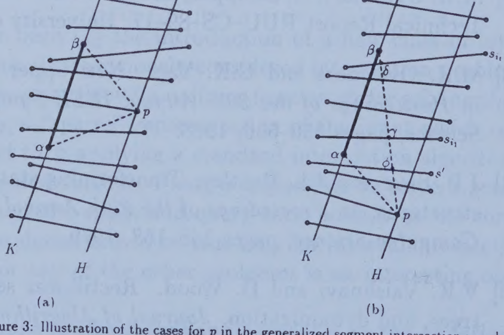

β
p
α
K
H
(a)
β
s_{i_t}
δ
γ
s_{i_1}
α
φ
s'
p
K
H
(b)

Figure 3: Illustration of the cases for p in the generalized segment intersection searching algorithm. The case symmetric to (b) is not shown.

On the Efficient Recognition of the Ellipse

Yu Chin Cheng Samuel C. Lee
School of EECS
University of Oklahoma
Norman, OK 73019

Abstract

Ellipses are of particular importance in pattern recognition since a wide range of man-made and natural objects take the shape of ellipses in an image obtained by projection. By studying properties of the parallel projection, we are able to derive a direct algorithm for ellipse recognition. Our algorithm is capable of extracting ellipses from an input point set that may contain other curves and isolated points caused by noise. Partial ellipses can also be extracted provided they contain curve segments that are sufficiently long. Theoretically, it has a time complexity of $O(n^3)$ and uses $O(m^2)$ space, where n is the number of points in the input and m^2 is the size of an array large enough to hold the input point set up to its resolution. The method proposed in the present work serves as a theoretical framework for practical algorithms to the recognition of ellipses in applications.

1 Introduction

Ellipses are of particular importance in pattern recognition since a wide range of man-made and natural objects take the shape of ellipses in an image obtained by projection. Extensive research has been done on recognition of ellipses, especially using transform methods. Such methods deal exclusively with algebraic equations of the ellipse, using the dual image of a point in the parameter space which is usually a high dimension hypersurface. Loci of the hypersurfaces are recorded in an accumulator array of the same dimension, and those of points which are on a same ellipse will intersect at a unique point in the parameter space. Then a count of the votes by each hypersurface is done, and those with high counts are identified. The coordinates of points with high counts in the parameter space represent the value of the coefficients of the equations of the ellipses. Limitations of the existing transform methods lie mainly in the excessive amount memory to store the accumulator array, and the time-consuming process to compute loci of hypersurfaces in the parameter space. These are particularly demanding when high accuracy is required.

In this paper, a theoretical framework using different representations of the ellipse is developed. By studying properties of the parallel projection, we are able to derive a direct algorithm for ellipse recognition. Basically, given three distinct points from an input scene that contains ellipses and other curves including isolated points which are considered noises, the algorithm costructs a hexagon that satisfies a set of properties of the parallel projection. By applying a matching condition that guarantees the correct recognition of an ellipse in the scene containing the vertices of the hexagon, it decides whether the triple of points can be used in the recovery of an ellipse in the input scene. When three points satisfy the matching condition, the ellipse interpolating them is constructed either by an implicit conic fitting or by a geometric method to be introduced. Partial ellipses can also be extracted provided they contain curve segments that are sufficiently long.

The rest of this paper is organized as follows. Section 2 introduces the notations and properties of the parallel projection used throughout the paper; Section 3 discusses a characterization of the ellipse that is of fundamental importance to our algorithm. In Section 4, a matching condition is derived. In Section 5, we present two representations of the ellipse and derive the related methods required by our algorithm. Section 6 presents the general algorithm for ellipse recognition. Finally, we summarize our present work in Section 7.

2 Preliminaries

The following notations are used throughout the paper.

- $\overleftrightarrow{AB}$, $\overrightarrow{AB}$, $\overline{AB}$, and $|\overline{AB}|$ denote the *straight line* or simply *line*, the *ray*, the *segment*, and the length of the segment determined by two *points* A and B.
- $\overset{\frown}{AB}$ denotes an arc on a circle or an ellipse determined by two points A and B on the circle or ellipse.
- $A_1A_2A_3...A_n$ denotes a *n-sided polygon* with vertices $A_1, A_2, A_3, ...,$ and A_n.

Given two distinct planes π and π' in space, π' is called the image of π under a parallel projection $T_{\mathcal{F}}$ if there is a family of parallel lines $\mathcal{F}$ in space not parallel to either plane that carries a point A in π into a point A' in π in a one-one fashion. A' is called the image of A under $T_{\mathcal{F}}$. Let l_1 and l_2 be two distinct lines in π, let A, B, and C be three distinct points on l_1 with images A', B', and C', respectively. The following lemmas are properties of a parallel projection [1]:

Lemma 2.1 *The image of l_1 in π is a straight line l'_1 in π'.*

Lemma 2.2 *If l_1 and l_2 are parallel in π, their respective images l'_1 and l'_2 are parallel in π'.*

Lemma 2.3 *Let C be a point which lies between A and B in l_1. Then C' lies between A', and B' in l'_1.*

Lemma 2.4 *If l_1 and l_2 in π are concurrent,i.e., they intersect at a point A, their respective images l'_1 and l'_2 remain concurrent and intersect each other at A' in π'.*

Lemma 2.5 *Let C be a point on $\overleftrightarrow{AB}$, $C \neq B$, with an image C' on the $\overleftrightarrow{A'B'}$. Then $C' \neq B'$ and $\frac{|\overline{AB}|}{|\overline{BC}|} = \frac{|\overline{A'B'}|}{|\overline{B'C'}|}$.*

Lemma 2.6 *The image of a circle $\mathcal{C}$ in π is an ellipse $\mathcal{C}'$ in π'. Conversely, for every ellipse $\mathcal{E}$ in a plane β, there exists a circle $\mathcal{C}$ in a plane α such that $\mathcal{E}$ and β are the images of $\mathcal{C}$ and α by a parallel projection.*

We also need the following property of a regular hexagon (see Figure 1.):

Lemma 2.7 *Let $ABCDEF$ be a regular hexagon inscribed in a circle $\mathcal{C}$. The center of $\mathcal{C}$, O, coincides with the intersection of $\overline{AD}$, $\overline{BE}$, and $\overline{CF}$. $ABCO$, $BCDO$, ..., etc. are parallelograms. Let G be the midpoint of $\overline{AB}$, let H be the intersection of $\overrightarrow{OG}$ with $\mathcal{C}$. $|\overline{OH}| : |\overline{OG}| = 2 : \sqrt{3}$.*

In the rest of this paper, we will omit mentioning the correspondence between a geometric object and its image as long as the meaning of such statement is clear from the context. Also, by an image we mean an image under a parallel projection.

3 A Characterization of the Ellipse

Consider inscribing a regular hexagon $ABCDEF$ in a circle $\mathcal{C}$. It is easy to see that $A'B'C'D'E'F'$ forms a hexagon in an ellipse $\mathcal{E}$ that is the image of $\mathcal{C}$ as a result of Lemma 2.1, Lemma 2.4, and Lemma 2.6 (see Figure 2.) Further, $\overline{A'B'} \parallel \overline{D'E'}$, $\overline{B'C'} \parallel \overline{E'F'}$, and $\overline{C'D'} \parallel \overline{F'A'}$ by Lemma 2.2. Finally, the center O' of $\mathcal{E}$ is determined by marking off twice the length $|\ \overline{B'G'}\ |$ from B' on the ray $\overrightarrow{B'G'}$, where G' is the midpoint of $\overline{A'C'}$. Note that D', E', and F' are symmetric about O' to A', B', and C', respectively. This construction follows from Lemma 2.5 and Lemma 2.7.

Therefore, if any three distinct points were the images of three adjacent vertices of a regular hexagon inscribed in a circle, center and three distinct symmetric points of the ellipse can be determined by a constant number of geometric constructions. Further, the symmetric points must also be in the ellipse. On the other hand, if any of these symmetric points were missing from the ellipse, the three given points cannot have been on a complete ellipse.

It is interesting to note that the hexagon vertices correspondence property stated above is an *existential* one; that is, we need not worry about the circle and its inscribed hexagon that existed, the mere knowledge of their existence is sufficient for us to claim that, on a ellipse, we must be able to find triples of three distinct points that are the images of three adjacent vertices of a regular hexagon.

We summarize the above property by the following theorem.

Theorem 3.1 (Weak Matching Condition) *Let A, B, and C be three distinct points on an ellipse. They are the images of three vertices of a regular hexagon inscribed in a circle whose image is the ellipse under a parallel projection if, and only if their symmetric points D, E, and F are also on the ellipse; further, such triples of points exist.*

It would seem that we have, by now, enough information to determine any ellipse containing the six points found in Theorem 3.1. For example, it is well-known that an ellipse can be determined by any five distinct points on the ellipse. However, it is important to realize that Theorem 3.1 can be used only as a *necessary condition* in a general setting. That is, for example, suppose there were two or more ellipses in the plane. It is quite possible that a six-tuple of inter-elliptic points satisfies the Weak Matching Condition. It would be an error to declare that an ellipse has been found. Such mis-characterizations seem inevitable since eventually, we are interested in processing images that contain curves other than ellipses and partial ellipses, and above all, points caused by noise.

In general, mis-characterizations may result in two ways. First, for a partial ellipse, although Theorem 3.1 allows us to compute three more points, they may not be in the image at all; it is likely that the partial ellipse cannot be found at all. Second, noisy points and inter-curve points may result in a great number of triples that satisfy Theorem 3.1 therefore obtaining ellipses that did not exist.

There are at least two ways to avoid mis-characterizations:

1. *Strengthen the Weak Matching Condition.* That is, for partial ellipses, we need points (other than the three symmetric points obtained by Theorem 3.1) on the ellipse that are *in* the image. Further, to avoid finding ellipses (that did not exist) caused by noisy points and inter-curve points, we may require more points that are on the ellipse determined by Theorem 3.1 to be also in the image. Both measures require that we be able to obtain more points from the six points obtained by the previous theorem in an efficient way.

2. *By verification.* With, for example, the equation of the ellipse found by using any five of the six points, we can easily verify whether a point in the image is on the ellipse. It is important that the equation be found in an efficient way.

It is intuitive to note that matching and verification are trade-offs to each other in the following sense: by extensive matching it is possible to eliminate the need of verification, and vice versa. Therefore, if matching is used alone, it must find a sufficient number of points to avoid mis-characterizations. On the other hand, although verification is fool-proof, using it alone would be highly inefficient. In fact, since there are C_3^n arrangements of triples of n points, the wcrst case complexity of an algorithm using verification along is $O(n^4)$. Same complexity applies if matching were done so extensively that all points on the supposed ellipse are computed. It is desirable that an algorithm take advantages of both the above measures.

In the most general setting, the ellipse extraction algorithm takes each triple of points in the image and finds a sufficient number of points required by an appropriate matching condition. If a match is found, extract the ellipse in an appropriate representation. Then verify the ellipse found is indeed in the picture. Once verified, all points in the ellipse found are eliminated from the input to avoid unnecessary processing. (Since there are other points on the ellipse that satisfy the matching condition, same ellipses would be found again.)

4 A Matching Condition

In this section, we shall derive a matching condition strong enough for a partial ellipse to be recognized. We begin with a definition and an immediate result from it (see Figure 3.)

Definition 4.1 *Let $\overline{AC}$ and $\overline{BD}$ be a pair of perpendicular diameters in a circle. Their image under parallel projection is called a pair of conjugate diameters.*

Theorem 4.2 *Let $ABCDEF$ be the hexagon and O be the center as stated in Theorem 3.1. Let G and H be the respective midpoints of $\overline{BC}$ and $\overline{EF}$. Mark off G'(resp. H') on $\overrightarrow{OG}$ (resp. $\overrightarrow{OH}$) so that $|\overline{OG'}| : |\overline{OG}|$ (resp. $|\overline{OH'}| : |\overline{OH}|$) $= 2 : \sqrt{3}$. Then $\overline{G'H'}$ and $\overline{AD}$ is a pair of conjugate diameters of the ellipse containing the hexagonal points.*

Proof: By Lemma 2.6, Lemma 2.7, and Definition 4.1 (see Figure 4.) ∎

It is a classical result form projective geometry that an ellipse can be constructed from a pair of conjugate diameters by using the ruler only [3]. In an analytical setting, this means we can compute the loci of an ellipse given the ability to compute intersection of two straight lines. We will present the classical result as a corollary of the following proposition.

Proposition 4.3 *Let $\overline{AC}$ and $\overline{BD}$ intersecting at the center O be a pair of perpendicular diameters of a circle C. Let $KLMN$ be the circumscribed square to C, where A, B, C, and D are the respective midpoints of $\overline{KL}$, $\overline{LM}$, $\overline{MN}$, and $\overline{NK}$. Let P be any point on the arc $\widehat{AD}$. Let X and Y be the intersections of $\overrightarrow{AP}$ with $\overline{KD}$ and $\overrightarrow{CP}$ with $\overline{OD}$, respectively. Then $\frac{|\overline{KX}|}{|\overline{KD}|} = \frac{|\overline{OY}|}{|\overline{OD}|} = \gamma$. Similar results hold in the other three circular arcs.*

Proof: Proof of the proposition is straightforward and is omitted. See Figure 5. ∎

By Lemmas 2.1 – 2.6, the classical construction reads Proposition 4.3 by replacing the words "circle" with "ellipse", "perpendicular" with "conjugate", and "square" with "parallelogram" (see Figure 5.)

Theorem 4.4 *See the text above.*

As a result of Theorem 4.4, we can construct an arbitrary number of points on the ellipse using different values of γ, $0 \leq \gamma \leq 1$, a desired property we pointed out earlier. In the case of a complete ellipse, this allows us to compute more points to avoid mis-characterization due to inter-curve and noisy points. For a partial ellipse that contains triples of points satisfying Theorem 3.1, although the symmetric triples may be missing from the input point set, we can always find a sufficient number of points on the curve segment in the vicinity of the triple granted that the input does contain such a partial ellipse. We summarize our finding in the following theorem.

Theorem 4.5 (Matching Condition) *Let A, B, and C be three distinct points on an ellipse or a partial ellipse. They are the images of three adjacent vertices of a regular hexagon inscribed in a circle whose image under a parallel projection is the complete ellipse, or contains the partial ellipse if, and only if sufficiently many points in the input can be found on the elliptic arcs $\widehat{AB}$ or $\widehat{BC}$.*

While Theorem 4.5 shows that a sufficient number of points are computed, it is desired that only a constant number of points are computed. Assume for the time being that the input contains only curves of second degree. Let C_1 be the would-be ellipse to be extracted in question, let C_2 be any other second degree curve, $C_2 \neq C_1$. Suppose the triple chosen in Theorem 4.5 are common to both C_1 and C_2. Bezout's theorem [4] says *if two curves of order two have more than $2 \times 2 = 4$ points of intersection, they represent an identical curve.* Now that we already have three distinct points on C_1, if C_1 did exist, by Theorem 4.5, two or more distinct points on $\widehat{AB}$ or $\widehat{BC}$ can be computed. Note that at most one of the additional points can be common to both C_1 and C_2; otherwise, $C_1 = C_2$, a contradiction to the assumption. On the other hand, if C_1 did not exist, by computing a sufficient number of points on the arcs $\widehat{AB}$ and $\widehat{BC}$, some computed points must be missing from the input. This shows mis-characterization can be avoided. Also, since the matching of the computed points against the input can be done in a constant time (that is, coordinates of the points computed are used as the indices to an two dimensional array which stores the input to its resolution,) Theorem 4.5 can be realized in a constant time.

5 Representations of the Ellipse

In this section, we shall study two representations that are of interest to us. Efficient methods to extract representations of the ellipse are shown.

5.1 Implicit Representation

A conic curve is represented by the loci of points (x, y) that satisfy the quadratic equation

$$Ax^2 + By^2 + Cxy + Dx + Ey + 1 = 0 \tag{1}$$

Equation 1 is called the *implicit representation* of a conic curve. Thus given the result of Theorem 3.1, we are in a position to determine a representation of the ellipse, i.e., resorting to conic fitting that uses any five of the six hexagonal points [2]. An elegant method due to R. A. Liming [2] requires only a few arithmetic operations (see Figure 6.) Let $l_1 = 0$, $l_2 = 0$, $l_3 = 0$ and $l_4 = 0$ be four linear equations. Let α be a real number, $0 \leq \alpha \leq 1$. Then

$$\alpha l_1 l_2 + (1 - \alpha) l_3 l_4 = 0 \tag{2}$$

is a quadratic equation which represents a conic curve.

Now let P, Q, R and S be four distinct points on an ellipse that are the intersections of l_1-l_3, l_2-l_3, l_2-l_4, and l_4-l_1, respectively. Equation (2) determines a family of conics through the four points. Let T be any other point on the ellipse, by substituting the coordinates of T into Equation (2) and solving for α, a quadratic equation for the ellipse is obtained.

The implicit representation enjoys great popularity among researchers in pattern recognition and geometric modeling, and has the advantage that it is quite easy to tell whether a point is on the curve. Verification is now straightforward; i.e., a point is on the ellipse found if it satisfies the equation.

5.2 Geometric Representation

An ellipse can be obtained from a circle by uniform compression along a diameter of the circle (see Figure 7.) More specifically, let $\overline{AC}$ and $\overline{BD}$ be two diameters perpendicular to each other in a circle $\mathcal{C}$. Let each chord on the circle parallel to $\overline{BD}$ be compressed to λ $(0 < \lambda \leq 1)$ its original length. That the result is an ellipse follows from the fact that uniform compression is a special form of an affine transformation which carries conics into conics [1], and that the compressed curve remains closed. In this case $\overline{AC}$ becomes the major axis of the resulting ellipse, and (the compressed) $\overline{BD}$ becomes the minor axis. Therefore, a *geometric* ellipse can be represented by the length of its major axis and a compression ratio. To fully specify its location and orientation in the Cartesian coordinate plane, we also need the center and the angle the major axis makes with the abscissa. We have the following definition.

Definition 5.1 *An ellipse in the Cartesian plane is uniquely determined by coordinates of the center, length of major axis, compression ratio, and the angle major axis makes with the abscissa.*

We now show how to extract the parameters as stated in Definition 5.1, given a pair of conjugate diameters. So far, we have obtained only the center. Toward this end, all we need is the coordinates of the end points of major and minor axes: compression ratio is obtained by computing the ratio of the length of minor axis to the length of major axis, and the rotation angle is obtained by taking the arc tangent of the slope of the major axis.

Let $\overline{AC}$ and $\overline{BD}$ be a pair of conjugate diameters of an ellipse with center O. Without loss of generality, assume $|\overline{AC}| \geq |\overline{BD}|$. If $\overline{AC}$ is perpendicular to $\overline{BD}$, $\overline{AC}$ (resp. $\overline{BD}$) must be the major (resp. minor) axis. In this case $|\overline{AD}| = |\overline{CD}|$ (see Figure 8.)

In general, $\overline{AC}$ will not be the major axis and $|\overline{AD}| \neq |\overline{CD}|$, see Figure 9. Without loss of generality, assume $|\overline{AD}| < |\overline{CD}|$. By the fact that major (resp. minor) axis is the longest (resp. shortest) diameter in an ellipse, there are two unique points M and N on the elliptic arc $\overset{\frown}{ADC}$ corresponding to one of the end points of major and minor axes such that $|\overline{ON}| \leq |\overline{OX}| \leq |\overline{OM}|$ for any other point X on the arc $\overset{\frown}{ADC}$. We claim that M lies in the arc $\overset{\frown}{AD}$ and N lies in the arc $\overset{\frown}{CD}$. To see this, produce a circle with center O and radius $|\overline{OD}|$ intersecting $\overline{OA}$ at the point P. It is easy to see that $|\overline{OX}| \geq |\overline{OP}| = |\overline{OD}| \geq |\overline{ON}|$. Therefore it is impossible for N to be in the arc $\overset{\frown}{AD}$. Similarly M cannot be in the arc $\overset{\frown}{CD}$ and the claim follows. This gives us an easy characterization on where to locate M and N.

Let M_1,M_2,and M_3 be three distinct points encountered in the arc $\overset{\frown}{AD}$ tracing from A to D. It is easy to see from the result of the previous paragraph that $|\overline{OM_2}|$ cannot be less than both $|\overline{OM_1}|$ and $|\overline{OM_3}|$. In fact, the following are the only possible cases:

1. $|\overline{OM_1}| \geq |\overline{OM_2}| \geq |\overline{OM_3}|$. Search for M is confined to $\overset{\frown}{AM_2}$.

2. $|\overline{OM_2}| \geq |\overline{OM_1}| \geq |\overline{OM_3}|$ or $|\overline{OM_2}| \geq |\overline{OM_3}| \geq |\overline{OM_1}|$. Search for M is confined to $\overset{\frown}{M_1M_3}$.

3. $|\overline{OM_3}| \geq |\overline{OM_2}| \geq |\overline{OM_1}|$. Search for M is confined to $\overset{\frown}{M_2D}$.

Note that M_1, M_2, and M_3 are chosen arbitrarily in the specified sense tracing from A to D. In particular, let M_1, M_2, and M_3 be the points of intersection of three pairs of lines corresponding to γ values 1/4, 2/4, and 3/4 in Theorem 4.4, respectively, it is easy to see that search range is reduced by half in terms of $|\overline{OD}|$ in each iteration. The search process may be terminated when the points on $\overline{OD}$ corresponding to the three γ values are sufficiently close to one another. M is then taken to be the point corresponding to the converging γ value. By symmetry, we can obtain the other end point of the major axis.

Search for N proceeds similarly, with the condition of case 1. becoming $|\overline{OM_1}| \leq |\overline{OM_2}| \leq |\overline{OM_3}|$; $|\overline{OM_2}| \leq |\overline{OM_1}| \leq |\overline{OM_3}|$ or $|\overline{OM_2}| \leq |\overline{OM_3}| \leq |\overline{OM_1}|$ for case 2; and $|\overline{OM_3}| \leq |\overline{OM_2}| \leq |\overline{OM_1}|$ for case 3.

It is easy to see that the search process converges in an exponential rate. We summarize our analysis in the following theorem.

Theorem 5.2 *Given a pair of conjugate diameters, the end points of the major axis and minor axis can be obtained by a search process that converges in an exponential rate.*

We like to remark that in a finite point set that represents the input picture (for example, Raster displays, digitized pictures, etc.,) there are only a finite number of points on $\overline{OD}$. In particular, suppose k points are on $\overline{OD}$, the search process in Theorem 5.2 terminates in $O(\log k)$ time.

6 The Algorithm

We are now ready to present the overall algorithm. Basically, we require that the ellipses or partial ellipses contain arcs long enough such that triples of points as stated in Theorem 4.5 exist. No other assumptions on the input are made.

The algorithm proceeds as follows. For all possible triples of points from the input set, perform a colinearity test. If the present triple is not colinear, Theorem 3.1 is used to obtain three more points and the center; Theorem 4.2 is used to obtain a pair of conjugate diameters. More points are obtained by Theorem 4.4 as required by the Matching Condition in Theorem 4.5. Once relevant points are found satisfying the Matching Condition, an implicit equation is determined by using Liming's method, or the parameters of an ellipse defined in Definition 5.1 are computed by the end points of major and minor axes extracted by Theorem 5.2. Finally, verify that there are indeed a sufficient number of points in the input. Once verified, the points on the ellipse found are deleted from the input. The algorithm is summarized in Table 1.

It remains to observe that (1) Colinearity test is necessary since the points in the input are not required to be in a *general position* (i.e., no three points are colinear.) This is important since exactly the same triple is computed as the symmetric set for a colinear triple by Theorem 3.1. (2) Since points on the extracted ellipses are deleted from the input, it is evident that, in later iterations, triples containing points that are already deleted need not be considered. (3) If the range of sizes of the ellipses in the input is known *a priori*, triples of points that are clustered together or too far apart from each other can be overlooked.

It is important to realize that the general algorithm serves only as a theoretical framework; for real applications, the nature of the problem must be carefully considered. For example, in image processing, the input usually contains pixel data which are integral coordinate pairs. When these integral pairs are used as a point set to our algorithm, they will at best represent only a sparse point set on the ellipse when real arithmetic is used. In a finite precision arithmetic, it is evident that rounding effect must be taken into account in carrying out the matching. In [5], a complete account on this problem is given. Theoretically, the algorithm has a complexity of $O(n^3)$ since we are guaranteed to find points on the ellipse in the vicinity of the triple, and the Matching Condition requires no more than a constant number of such additional points. It uses no more than $O(m^2)$ space where the m^2 space is the size of the array large enough to hold the quantized display screen or digitized picture containing the input point set.

7 Conclusion

A theoretical framework for the correct and efficient recognition of the ellipse is proposed. By the properties of the ellipse as an image of a circle under a parallel projection, a direct algorithm is derived. The algorithm has a time complexity of $O(n^3)$ and uses $O(m^2)$ space, where n is the number of points in the input and the m^2 space is used by an array large enough to hold the input point set up to its resolution. We are currently extending the theoretical algorithm to derive both time- and space- efficient algorithms to the recognition of ellipses in real applications.

References

[1] D. Gans, *Transformations and Geometries*, Appleton-Centry-Crofts, New York, 1969.

[2] M. A. Penna and R. R. Patterson, *Projective Geometry and its Applications to Computer Graphics*, Prentice Hall, New Jersey, 1986.

[3] T. F. Holgate, *Projective Pure Geometry*, Macmillan, New York, 1930.

[4] R. J. Walker, *Algebraic Curves*, Princeton University Press, Princeton, New Jersey, 1950.

[5] Y. C. Cheng and S. C. Lee, "New Algorithms for Ellipse Recognition", in preparation.

* * *

- **Input:** A set of n points in the plane.
- **Output:** Implicit or geometric representation of ellipses in the input that satisfy Theorem 4.5.
- **Algorithm:**

```
BEGIN
    FOR all possible triple of points DO
        BEGIN
            IF they are not co-linear THEN
                BEGIN
                    Compute three additional points by Theorem 3.1;
                    Find a pair of conjugate diameters by Theorem 4.2;
                    Compute the required points by Theorem 4.4;
                    IF points found satisfy the Matching Condition THEN
                        BEGIN
                        Implicit representation:
                            Use Liming's method to determine the equation;
                            Verify by substituting all points to the equation;
                        OR
                        Geometric representation:
                            Find end points of major and minor axes by Theorem 5.2;
                            Compute all parameters required by Definition 5.1;
                            Verify by reconstructing the ellipse extracted;
                            Delete all points on the ellipse extracted;
                        END
                END
        END
END.
```

Table 1: General Algorithm for Ellipse Extraction

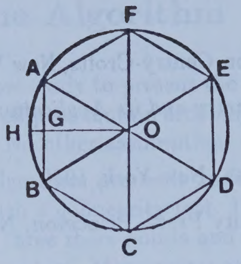


Figure 1.

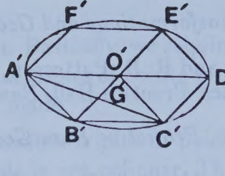


Figure 2.

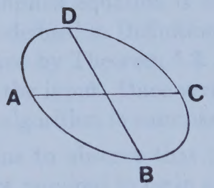


Figure 3.

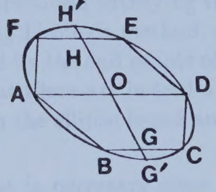

H′
F
E
H
O
A
D
G
C
B
G′

Figure 4.

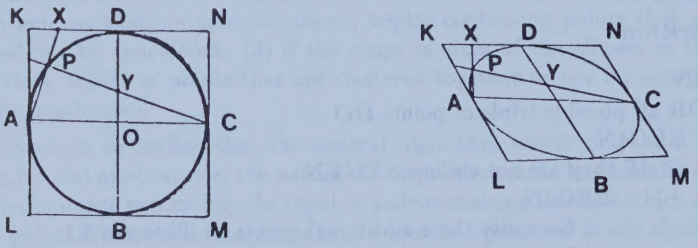

K
X
D
N
P
Y
A
C
O
L
B
M
K
X
D
N
P
Y
A
C
L
B
M

Figure 5.

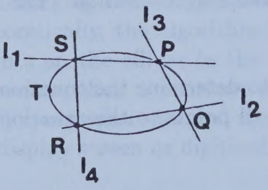

I3
I1
S
P
T
I2
Q
R
I4

Figure 6.

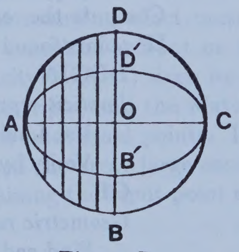


Figure 7.

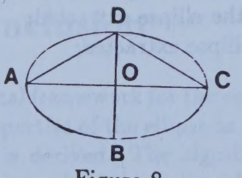


Figure 8.

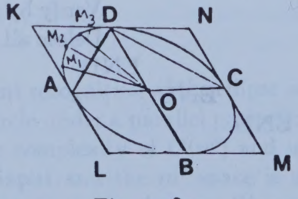

K
M3
D
N
M2
M1
A
C
O
L
B
M

Figure 9.

2-Concave Rectilinear Polygons - Necessary and Sufficient for Graph Dualization[1]

K. H. Yeap and *M. Sarrafzadeh*[2]
Department of Electrical Engineering and Computer Science
Northwestern University
Evanston, IL 60208

Abstract

Given a planar triangulated graph (PTG) G, we study the problem of constructing a rectilinear floor-plan F (boundary of F is rectangular) such that G is the dual of F. We show that if only 0-concave rectilinear polygons (CRP) (i.e., rectangles) and 1-CRP (i.e., L-shaped) are allowed, there are PTG's that do not admit any dual. However, if 2-CRP (e.g., T-shaped and Z-shaped) are also allowed, then every PTG admits a dual. A linear-time algorithm for constructing a 2-CRP dual of an arbitrary PTG is proposed.

1 Introduction

The most immediate representation of a floor-plan (FP) is the partition of a rectangular area into polygons (usually rectilinear). Such a partition can be represented by a partition graph where vertices represent junctions and edges represent the sides of polygons. The dual graph of the partition is a graph where each vertex represents a polygon and each edge (i, j) represents adjacency of polygon i and polygon j. Given a partition, its dual graph is easily and uniquely determined. However, given a graph specifying adjacency requirements, it is not readily converted into a partition. In this formulation of floor-planning problem, we are given a graph G and want to find a rectilinear partition P on a rectangular area such that G is the dual of P.

Most formulations incorporate only 0-concave rectilinear polygons (CRP), i.e., rectangles. Other generalizations allow more complicated shapes. In this paper we examine 0-CRP, 1-CRP and 2-CRP. 1-CRP (2-CRP) refers to a rectilinear polygon that consists of 6 (8) sides, 5 (6) convex and 1 (2) concave corners. A 1-CRP is also called L-shaped due to its appearance. There is only one type of polygon for 0-CRP and 1-CRP if we disregard polygon orientations. A 2-CRP can be viewed as two adjacent 1-CRP. There are four classes of 2-CRP as shown in Figure 1.

[1] and [3] have developed the necessary and sufficient conditions for the existence of a 0-CRP dual. [4] reported a liner algorithm for constructing a 0-CRP dual if one exist.

---

[1] this work was supported in part by NSF under grant MIP-8921540

[2] E-mail: yeap@eecs.nwu.edu, majid@eecs.nwu.edu

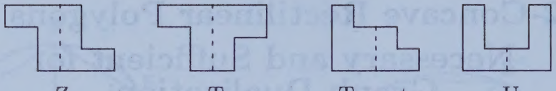


Figure 1: Four classes of 2-CRP: Z, T, Two-step and U.

The construction of a dual is complicated by the existence of complex triangles (CT's) (a cycle of three edges that is not a face) because the dual of a rectangular partition does not contain any CT. One approach eliminates all CT's to obtain a rectangular floor-plan [5]. This technique introduces new vertices and edges to the original PTG, producing empty spaces in the floor-plan. If the empty spaces are considered part of some adjacent polygons, this approach produces general rectilinear shaped polygons.

Adjacency requirements of CT's can be achieved by introducing more complicated shapes instead of rectangle. A 1-CRP with 1-bend can satisfy the adjacency requirements of a CT. Similarly a 2-CRP can satisfy the adjacency requirements of two CT's simultaneously. In some sense, the introduction of more complicated shapes is analogous to eliminating complex triangles in a PTG. However, this approach maintains the adjacency requirement without arbitrarily adding new vertices and edges to the input PTG. Our approach ensures that the shape of the output polygons will be confined to the specific shapes we introduce (0, 1 and 2-CRP) and will not produce arbitrary shaped rectilinear polygons nor any empty spaces.

The necessary and sufficient conditions for the existence of an 1-CRP dual of a PTG are given in [2]. However, there exist PTG's, as we will show, where 1-CRP dual is not sufficient. Obviously, an arbitrary PTG admits a dual if we allow general rectilinear polygons. The problem of constructing a dual using k-CRP for some constant k was open. We show that any planar triangulated graph admits a 2-CRP dual. A linear-time algorithm for constructing a 2-CRP dual is presented. We thus establish that employment of 2-CRP is both necessary and sufficient for floor-planning using graph dualization technique.

This paper is organized as follows: Section 2 discusses the major issues arising in the dual construction of a PTG, especially the local adjacency requirements. Section 3 presents a key theorem for the establishment of the necessary and sufficient conditions of a dual. An algorithm for dual construction is proposed in section 4. All proofs, detail descriptions and extensions are omitted in this summary paper.

2 Graph Dualization

In this section, we discuss the major issues arising in graph dualization of a PTG. We examine the local adjacency requirements of a PTG and show that 1-CRP is not sufficient. The notion of sharing, which is crucial to dual construction, is also discussed.

2.1 Terminologies

A *planar triangulated graph (PTG)* is a bi-connected planar graph that has only triangulated faces except the infinite face. A vertex of a PTG that is located on the infinite face is called an *external vertex.* An *extended planar triangulated graph (EPTG)* G_e is a special case of PTG which is obtain by adding 4 vertices, l, u, r, b, and some edges to a PTG, $G = (V, E)$, where:

$$\begin{aligned} G_e &= (V \cup \{r,u,l,b\}, E \cup E_r \cup E_u \cup E_l \cup E_b) \\ E_r &= \{(r,v) | v \in V_r\} \\ E_u &= \{(u,v) | v \in V_u\} \\ E_l &= \{(l,v) | v \in V_l\} \\ E_b &= \{(b,v) | v \in V_b\} \end{aligned}$$

V_r, V_u, V_l and V_b is respectively the set of external vertices of G that is adjacent to r, u, l and b. The added vertices r, u, l, and b represent four fictitious rectangles used to enclose G in a rectangular area. Edges in E_r, E_u, E_l and E_b represent adjacency of polygons of G with the fictitious rectangles. A non-trivial EPTG consists of at least 5 vertices. Figure 2 shows a construction of an EPTG from a planar graph G.

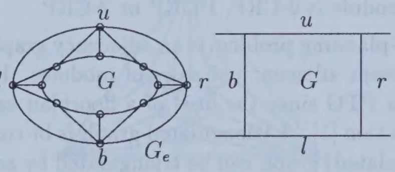


Figure 2: Construction of an EPTG G_e from a planar graph G.

A *path* is an ordered set of vertices $(v_1, \ldots, v_n)$ where $v_1, \ldots, v_n$ are mutually distinct. A *cut* is a path $(u, x_1, \ldots, x_n, v)$ in a PTG where u and v are external vertices and $x_1, \ldots, x_n$ are not external vertices of the PTG. Let P be a cut which decomposes a PTG into two components G_1 and G_2. We say that P *cuts* PTG into G_1 and G_2. Vertices and edges of P appear in both G_1 and G_2. A *minimal path* M is a path, where every non-adjacent pair (u, v) of M is not adjacent in PTG. If a minimal path is a cut, we call it a *minimal cut.* Given a path P in a PTG, we can easily construct a minimal path from P. Such a minimal path is denoted as P^*.

A *complex triangle (CT)* is a non-face cycle of a PTG containing exactly three edges. The set of all complex triangles in a planar graph G is denoted by $CT(G)$. A CT is denoted as $\triangle$. A CT formed by vertices a, b, c and edges $(a, b), (a, c)$ and (b, c) is denoted as $\triangle_{abc}$. Figure 3 shows two examples of complex triangles.

An assignment on vertex v is a pair $(\triangle, v)$ where $\triangle$ is a CT and v is a vertex of $\triangle$. We say that the CT $\triangle$ is *assigned* to v. A vertex v which has been assigned by two CT's is said to be *saturated*, otherwise it is *unsaturated.* A *perfect assignment of a PTG* is a set of assignment of all CT's of the PTG where:

1. Every CT is assigned to a vertex.

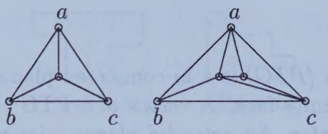


Figure 3: Complex triangles

2. No vertex carries more than two assignments.

A *perfect assignment of an EPTG* is a perfect assignment with additional condition that:

3. Boundary vertices of G_e (i.e., r, u, l and b) are not assigned.

In a perfect assignment, a vertex of a PTG could be assigned 0, 1, or 2 times.

2.2 0-CRP, 1-CRP and 2-CRP Dual

A *0-CRP dual* of a graph G is a partition P of a rectangular area where each module is a 0-CRP and the corresponding adjacency graph of P is G. Similarly, we define *1-CRP dual* to be a partition where each module is either a 0-CRP or 1-CRP. A *2-CRP dual* is a partition where each module is 0-CRP, 1-CRP or 2-CRP.

The input of the floor-planning problem is an adjacency graph where vertices represent modules and edges represent adjacency of pairs of modules. It is reasonable to assume that the input graph is a PTG since the dual of a floorplan can always be made into a PTG by a minor modification [1]. A triangulated graph is bi-connected by our definition. A planar (but not triangulated) graph can be triangulated by adding edges. An arbitrary graph can be triangulated by adding edges and/or vertices.

If an input graph contains CT's, the adjacency requirements cannot be satisfied by 0-CRP dual [1]. A 1-CRP with one concave corner is able to realize adjacency requirement of one CT (see Figure 4). Similarly, a 2-CRP can satisfy adjacency requirement of two CT's simultaneously (see vertex 3 of Figure 4). The adjacency requirements of all CT's must be satisfied in a dual. In our formulation, 0-CRP, 1-CRP and 2-CRP correspond to vertices which are assigned 0, 1 and 2 times (respectively) in a perfect assignment.

2.3 Insufficiency of 1-CRP Duals

Given an arbitrary PTG, some vertices of a CT must be 1-CRP or 2-CRP to satisfy the local adjacency requirements. [1] proved the necessary and sufficient conditions for a graph to admit a 0-CRP dual. It was further generalized by [2] to prove the necessary and sufficient conditions for the existence of a 1-CRP dual. The primary condition for the existence of a 1-CRP dual is that all non-overlapping (i.e., one is not contained in the other) CT's can be assigned to some vertices and each vertex has no more than one assignment. Given a PTG, they construct an EPTG G_e and assign CT's to vertices. If a vertex is assigned, it becomes a 1-CRP in the corresponding floor-plan; otherwise it is a rectangle. However, not all PTG's admit 1-CRP dual. In particular, consider the PTG shown in Figure 5: the graph contains 7 non-overlapping CT's and only 6 vertices

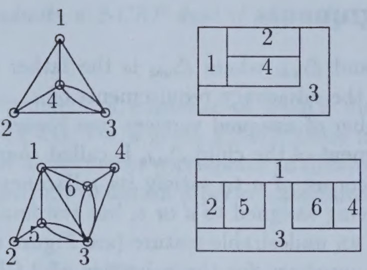


Figure 4: Floor-plans of complex triangles.

are available for assignment. There is no assignment that can satisfy the necessary conditions for the existence of a 1-CRP dual. This suggests that the dual of an arbitrary PTG requires more complicated shapes. We prove that generalization to 2-CRP dual is necessary and sufficient for the existence of a dual of any PTG.

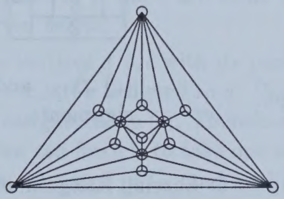


Figure 5: A PTG that does not have any 1-CRP dual.

2.4 Complex Triangle Graph and Containment Tree

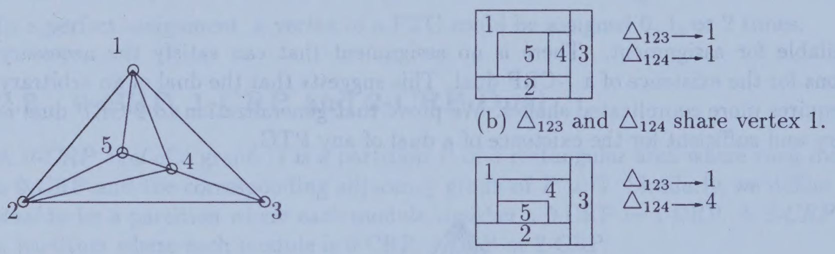
Let $G = (V, E)$ be a PTG. A complex triangle graph (CTG) of G, is $CTG(G) = (V_c, E_c), V_c \subseteq V$ and $E_c \subseteq E$ where:

$$V_c = \{\ v \mid v \text{ belongs to some CT's of } G\ \}$$
$$E_c = \{\ e \mid e \text{ belongs to some CT's of } G\ \}$$

Given a PTG it is easy to construct a CTG thereof. A CT is always assigned to a vertex v of V_c. Therefore, it suffices to consider $CTG(G)$ in search of a perfect assignment of G. There is a natural hierarchical structure among the CT's of a CTG. The tree representing the hierarchical relationship is called a *containment tree.* A pseudo root node is added as the parent of first level CT's if needed. Traversing the tree top-down allows us to process the CT's hierarchically.

2.5 Sharing Assignments

Consider two CT's $\triangle_{abc}$ and $\triangle_{ade}$ where $\triangle_{abc}$ is the father of $\triangle_{ade}$. Suppose $\triangle_{abc}$ is assigned to a. To satisfy the adjacency requirements of $\triangle_{ade}$, we can naturally assign it to a to reduce the number of assigned vertices (see Figure 6b, $\triangle_{abc} = \triangle_{123}$, $\triangle_{ade} = \triangle_{124}$, $a = 1$). Such assignment of the child $\triangle_{ade}$ is called *sharing assignment* since $\triangle_{ade}$ shares the same concave corner of a to satisfy its adjacency requirements. Certainly, $\triangle_{ade}$ has the freedom of being assigned to d or e, but the final floor-plan will have more non-rectangular polygons, an undesirable feature (see Figure 6c, vertex 1 and 4). It can be shown that sharing is necessary for the existence of 2-CRP duals. An assignment algorithm must incorporate sharing scheme to achieve a perfect assignment.


1
5
4
2
3
1
5
4
3
2
$\triangle_{123} \rightarrow 1$
$\triangle_{124} \rightarrow 1$
1
4
3
5
2
$\triangle_{123} \rightarrow 1$
$\triangle_{124} \rightarrow 4$

(a) A planar triangulated graph.

(b) $\triangle_{123}$ and $\triangle_{124}$ share vertex 1.

(c) $\triangle_{123}$ and $\triangle_{124}$ have independent assignments.

Figure 6: Sharing Assignment.

When the child CT $\triangle_{ade}$ shares its assignment with its parent, it does not add more bend to polygon a. Therefore, we do not increase the assignment count on vertex a when sharing occurs. All descendants of a CT can share its assignment as long as they have a common vertex.

3 2-CRP Duals

A 2-CRP dual is constructed by recursive subdivision based on a minimal cut that decompose G into G_1 and G_2. Such a partition has the properties stated in Lemma 1.

Lemma 1 *Let $P^* = (w_1, \ldots, w_n)$ be a minimal cut in a PTG G and G_1, G_2 are two sub-graphs cut by P^*, then:*

1. $CT(G) = CT(G_1) \cup CT(G_2)$ *and,*
2. $CT(G_1) \cap CT(G_2) = \phi$.

Intuitively, Lemma 1 says that the set of CT's in G can be decomposed into two disjoint sets by a minimal cut. Furthermore, such a decomposition preserves the "integrity" of each CT in G, i.e., a CT of G must be found in either G_1 or G_2, exclusively. The lemma also suggests a method of recursive construction of 2-CRP dual from a PTG, which leads to our main theorem:

Theorem 1 *A PTG G admits a 2-CRP dual if and only if an EPTG G_e of G has a perfect assignment.*

From theorem 1, the problem of existence of 2-CRP dual is reduced to finding a perfect assignment in an EPTG. For a given EPTG, there may be more that one perfect assignments. Different assignments correspond to different floor-plans. In the next section, we present an algorithm to obtain a perfect assignment in an EPTG.

4 An algorithm for Perfect Assignment

The hierarchical structure of CT suggests a recursive assignment procedure. At each step of the recursion, we only consider the CTG G_i where the CT's of G_i of are children of a node $\triangle_{abc}$ in the containment tree. We assume that G_i is connected. If not, our algorithm will be applied to each connected component of G_i. After G_i is assigned, the algorithm is recursively applied to the children of $\triangle_{abc}$ in the containment tree.

We define the following terms on G_i: an edge (a, b) is called an *internal edge* if there exist triangles $\triangle_{abc}$ and $\triangle_{abd}$ ($c \neq d$) in G_i; otherwise, it is called a *boundary edge.* A vertex is called a *boundary vertex* if there are some boundary edges incident to it; otherwise, it is called an *internal vertex.*

G_i can share at most three vertices a, b, c with its parent $\triangle_{abc}$. In the worst case, all of them may be saturated. Let $\triangle_{abc}$ be assigned to a. Due to the sharing scheme, vertex a is allowed to take one more assignment in G_i. Therefore, even if a is already saturated from its parent's assignment, we can force it to take one more assignment in G_i. Thus at most two vertices (b and c) are saturated in G_i, and both of them are boundary vertices.

As the algorithm proceeds, we delete edges and vertices of G_i which do not have unassigned triangles incident to them. We always maintain the following invariant properties:

P1. At most two boundary vertices are saturated. The rest of the boundary vertices may have 0 or 1 assignment.

P2. Internal vertices have zero assignment.

We choose an unsaturated vertex v from the set of boundary vertices. Such a vertex always exist since a non-trivial G_i has at least three boundary vertices. We scan the vertices adjacent to v counter-clockwise. Let $v_0, \ldots, v_n, n \geq 1$ be the vertices where (v, v_0) and (v, v_n) are boundary edges. (Note that v_0 and v_n are boundary vertices). There are at most n triangles $\triangle_1, \ldots, \triangle_n$ incident to v, where $\triangle_i$ is $\triangle_{v_{i-1}\, v_i\, v}$. There are two possible cases:

CASE 1 $v_1, \ldots, v_{n-1}$ are all internal vertices.

CASE 2 Some of $v_1, \ldots, v_{n-1}$ are boundary vertices.

CASE 1 $v_1, \ldots, v_{n-1}$ *are all internal vertices.*

We will assign $\triangle_i$ to $v_i, i = 1, \ldots, n-1$ and assign $\triangle_n$ to v. We delete vertex v and edges $(v, v_0), \ldots, (v, v_n)$ since all incident triangles have been assigned. $v_0, \ldots, v_n$ is added to the set of boundary vertices. The resulting graph is smaller and the invariant

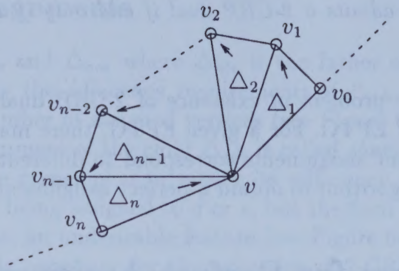


Figure 7: Assignment when $v_1, \ldots, v_{n-1}$ are internal vertices.

properties **P1**, **P2** are maintained. The vertex v, which has been deleted, is assigned at most twice. The procedure is shown in Figure 7. The arrows in Δ_i indicate the assignments.

CASE 2 *Some of* $v_1, \ldots, v_{n-1}$ *are boundary vertices.*

In this case, we will not make any assignment. Instead, we will cut G_i into two smaller sub-graphs where the invariant properties **P1**, **P2** are maintained in both sub-graphs. Let $v_m (1 \leq m \leq n-1)$ be the boundary vertex with the smallest index m. Consider the edge (v, v_m) where v and v_m are boundary vertices. The edge (v, v_m) decomposes G_i into two sub-graphs G_{ir} and G_{il} with (v, v_m) appearing in both sub-graphs. There are two cases:

CASE 2.1 v_m is saturated.

CASE 2.2 v_m is not saturated.

CASE 2.1 v_m *is saturated.*

There are only two saturated vertices in G_i and by our selection, v is not saturated. Without loss of generality, we can assume that the other saturated vertex v_s is in G_{il}. We *force* vertex v in G_{ir} to be saturated, thus denying triangles in G_{ir} to be assigned to v. G_{ir} has exactly two saturated vertices: v_m and v (forced saturation), so has G_{il}: v_m and v_s. Vertex v in G_{il} will keep whatever assignment it carries in G_i but it will not be assigned by triangles in G_{ir}. Thus vertex v in G_i will not be over-saturated due to forced saturation of v in G_{ir}. The decomposition is depicted in Figure 8.

CASE 2.2 v_m *is not saturated.*

Let the two saturated vertices be v_{s1} and v_{s2}. v_{s1}, $(v_{s2}) \in G_{ir}$ if and only if v_{s1}, $(v_{s2}) \notin G_{il}$ because the vertices v and v_m, which are the only vertices in both sub-graphs, are not saturated. If $v_{s1} \in G_{ir}$ and $v_{s2} \in G_{ir}$, we force v and v_m of G_{il} to be saturated. If $v_{s1} \in G_{ir}$ and $v_{s2} \in G_{il}$, we force v of G_{il} and v_m of G_{ir} to be saturated. If $v_{s1} \notin G_{ir}$ and $v_{s2} \notin G_{ir}$, we force v and v_m of G_{ir} to be saturated. In any case, G_{ir} and G_{il} have exactly two saturated vertices. v (v_m) of G_{il} is saturated if and only if v (v_m) of G_{ir} is not saturated. Thus v and v_m will not be over-saturated in G_i. The decomposition is shown in Figure 9.

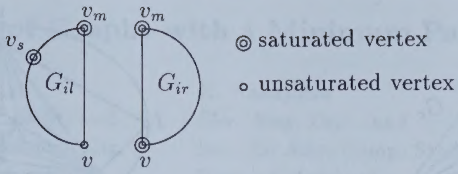
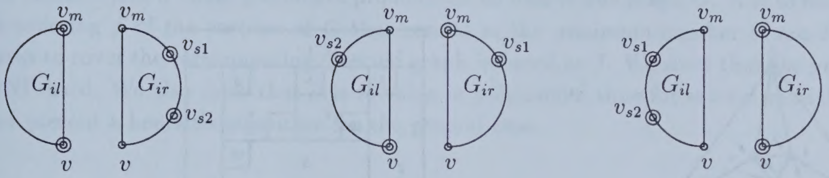


Figure 8: Decomposition when v_m is saturated.



(a) $v_{s1} \in G_{ir}, v_{s2} \in G_{ir}$.

(b) $v_{s1} \in G_{ir}, v_{s2} \in G_{il}$.

(c) $v_{s1} \notin G_{ir}, v_{s2} \notin G_{ir}$.

Figure 9: Decomposition when v is not saturated.

From the above discussion, we either eliminate some triangles or decompose G_i into smaller sub-graphs. Thus we have an assignment algorithm on G_i with the invariant property as inductive hypothesis. If G_i has only one triangle, the assignment problem is trivial.

Data structures that support the algorithm are relatively simple. A doubly connected edge list [6] can be used to find CT's incident to a vertex. The algorithm visits each CT without backtracking. Therefore the complexity of the assignment algorithm is $O(c)$, where c is the number of CT. c is $O(|V|)$ in a planar graph. Therefore the overall time complexity is $O(|V|)$ and we have the following theorem:

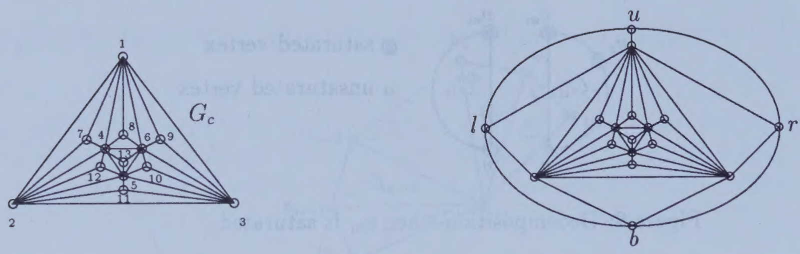
Theorem 2 *Given a PTG, a perfect assignment can be constructed in $O(|V|)$ time where $|V|$ is the number of vertices of the PTG.*

Theorem 2 can be easily modified to produce assignment for EPTG. The only restriction is to avoid assignment to the four external vertices of an EPTG. Thus we have the following theorem:

Theorem 3 *Given an EPTG constructed from a PTG, a perfect assignment can be constructed in $O(|V|)$ time where $|V|$ is the number of vertices of the PTG.*

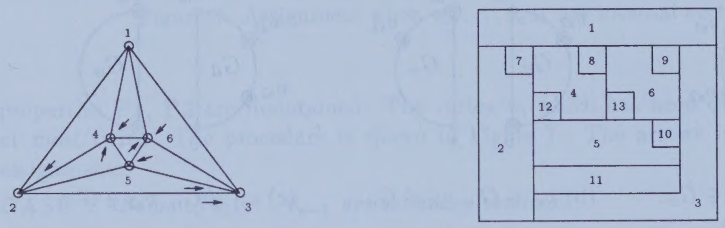
Example:

We demonstrate an example of the construction process using the graph of Figure 5, which does not admit any 1-CRP dual. The results are shown in Figure 10. (a) is the PTG and (b) shows an EPTG. (c) is a perfect assignment and (d) gives the 2-CRP dual floor-plan constructed with the perfect assignment in (c).



(a) Input Graph G.

(b) EPTG constructed from G.



(c) A perfect assignment.

(d) Floor-plan constructed from the perfect assignment in (c).

Figure 10: An example of dual construction.

References

[1] K. Kozmonski and E. Kinnen, "Rectangular Dual of Planar Graphs," *Networks,* vol. 15, pp. 145-157, Jan. 1985.

[2] Y. Sun and M. Sarrafzadeh, "Floorplanning by Graph Dualization: L-Shaped Models," *Proceeding of the IEEE International Symposium on Circuits and Systems,* New Orleans, LA, pp. 2845-2848, 1990.

[3] Y.T. Lai and S.M. Leinwand, "Algorithms for Floor-plan Design via Rectangular Dualization," *IEEE Transactions on Computer Aided Design,* vol. 7, no. 12, pp. 1278-1289, Dec. 1988.

[4] J. Bhasker and S. Sahni, "A Linear algorithm to Find a Rectangular Dual of a Planar Triangulated Graph," *Algorithmica,* vol. 3, no. 2, pp. 247-278, 1988.

[5] S. Tsukiyama, K. Koike, and I. Shirakawa, "An Algorithm to Eliminate All Complex Triangles in a Maximal Planar Graph for Use in VLSI Floor-plan," *Proceeding of the IEEE International Symposium on Circuits and Systems,* San Jose, CA, pp. 321-324, 1986.

[6] F. Preparata and M. Shamos, *Computational Geometry - An Introduction,* Springer-Verlag, 1985.

Linear Placement of Graphs with a Minimum Path Cover [1]

P.-Y. F. Lin
Elec. Eng. Dept. and
Systems Research Ctr.
Univ. of Maryland
College Park, MD 20742

K. Nakajima
Elec. Eng. Dept. and
Inst. for Adv. Comp. Studies
Univ. of Maryland
College Park, MD 20742

Abstract

We consider a new linear placement problem for an undirected graph G. It is to find such an ordering f of the vertices of G that results in the minimum number of arc-disjoint paths to cover the corresponding directed graph induced by f. We show that the problem is NP-hard. We also show that it is solvable in polynomial time for several special cases and present a heuristic algorithm for the general case.

1 Introduction

Consider an undirected graph $G = (V, E)$ and a 1-1 mapping $f : V \rightarrow \{1, 2, \ldots, |V|\}$, which we call a *linear layout* of G. Without any loss of generality, we assume in this paper that all graphs are connected. Orienting each edge $(u, v) \in E$ as an arc $\langle u, v\rangle$ if and only if $f(u) < f(v)$, we obtain a directed graph, which we call the *f-induced orientation* of G and denote by $\vec{G}_f$. Two directed paths are said to be *arc-disjoint* if they do not share an arc. A path cover of a directed graph is a set of arc-disjoint directed paths that contain all arcs of the graph and is said to be minimum if the number of such paths is the smallest. In this paper, we consider the problem of finding a linear layout f of G that requires a minimum path cover of the corresponding $\vec{G}_f$. The corresponding decision problem is formally described below.

Minimum Arc-disjoint Path Cover (MPC)
Instance: An undirected graph $G = (V, E)$ and a positive integer $K \leq |E|$.
Question: Is there a linear layout f of G such that its f-induced orientation $\vec{G}_f$ has a path cover containing $\leq K$ arc-disjoint directed paths?

This problem may be reformulated as follows. Let $H(U, A)$ be a directed graph. For each $u \in U$, let $DD(u) = |d_o(u) - d_i(u)|$, where $d_o(u)$ (resp., $d_i(u)$) is the out-(resp., in-)degree of vertex u in H. We call DD(u) the *Degree-Difference-value* or in short, *DD-value* of u. The sum of DD-values of all vertices in H is called its *total DD-value* and it is always a positive even integer. Suppose that a linear layout f of G is given. It can be seen that the number of arc-disjoint directed paths that cover $\vec{G}_f$ is equal to one half of its corresponding total DD-value (see Fig. 1(b)). Hence, the problem of minimizing the number of such paths is equivalent to that of minimizing the total DD-value. This latter problem is formally defined as follows.

Minimum Total DD-value (MTD)
Instance: An undirected graph $G = (V, E)$ and a positive even integer $K' \leq 2\cdot|E|$.

---

[1] This work was supported in part by National Science Foundation Grants MIP-84-51510 and CDR-88-03012 and in part by a grant from Minta Martin Fund for Aeronautical Research, University of Maryland, College Park, MD.

Question: Is there a linear layout f of G such that its f-induced orientation $\vec{G}_f$ has a total DD-value $\leq K'$?

In its practical aspects, the MPC problem may be related to the automatic generation of *Metal-Metal Matrix* (M^3) layouts of high speed CMOS circuits [3, 6]. In such layouts, we want to merge transistors by diffusion abutments into as few diffusion columns as possible in order to reduce the width of the layout. In this situation the nets must be properly ordered in the diffusion columns so as to avoid conflicts of the routing of nets that connect two or more diffusion columns on one layer of metal. Given a dynamic CMOS circuit, we model its nMOS network by a graph G in which each edge represents a transistor and each vertex a net (see Fig. 1(a)). Then, the transistor merging problem may be related to the problem of finding a linear layout f of G that results in a minimum path cover of $\vec{G}_f$ (see Fig. 1(b)).

In its theorectical aspects, MPC is a linear placement problem of graphs which employs a new objective function; i.e., the number of arc-disjoint directed paths that cover their corresponding directed graphs. The objective functions of most of the other linear placement problems studied in the past are related to edge lengths and cuts defined as follows. Given a graph $G = (V, E)$ and an f, the *length* of an edge $e = (u, v)$ is defined as $|f(u) - f(v)|$ and denoted by $len(e|f)$ and the *cut* of G at position i of f is defined as $|\{(u, v) \in E | f(u) \leq i < f(v)\}|$, which is denoted by $cut(i|f)$. The optimal linear arrangement problem which asks to minimize $\sum_{e\in E,f} len(e|f)$ has been shown to be NP-hard for general graphs [5] and polynomially solvable for trees [11] and outerplanar graphs [2]. The min-cut problem which asks to minimize $\max_{i,f}\{cut(i|f)\}$ has also been proved to be NP-hard even for graphs with maximum degree 3 [9] and polynomially solvable for trees [13] and outerplanar graphs [2]. The bandwidth minimization problem which asks to minimize $\max_{e\in E,f}\{len(e|f)\}$ has been shown to be NP-hard even when G is restricted to caterpillars with hair length 3 [10] and polynomially solvable for interval graphs [7] and outerplanar graphs [2]. Some variations of these problems are discussed in [8, 9]. Recently, Simonson [12] studied a different type of linear placement problem. In his problem, certain 'critical' paths, say $(p_1, p_2, \ldots, p_k)$, (not necessarily edge-disjoint) have specified length constraints $(b_1, b_2, \ldots, b_k)$ where b_i's are positive integers. The problem is to find a linear layout of G such that $\sum_{e\in p_i}\{len(e|f)\} \leq b_i$ for each path p_i.

In this paper, we show that problem MPC is NP-complete even when G is a bipartite planar graph in Sec. 2. In Sec. 3, we show that the optimization version of MPC is solvable in linear time when G is restricted to a tree or a grid graph. We also show that when K' is fixed, it is solvable in polynomial time for any graph. Finally, in Sec. 4, we develop a greedy heuristic algorithm for the general case of the optimization version and provide some experimental results.

2 The NP-Completeness

We show the NP-completeness of MTD by a polynomial transformation from the *exact cover by 3-sets (X3C) problem* [4], which is defined as follows.

Exact Cover by 3-Sets (X3C)

Instance: A set X of $3q$ elements and a collection C of 3-element subsets of X.

Question: Does C contain a subcollection C' such that every element of X occurs in exactly one member of C'?

Problem X3C remains NP-complete even when each $x \in X$ is contained in two or three subsets in C [4].

Probelm MTD is clearly in NP since by guessing a linear order and then computing its total DD-value, MTD is nondeterministically solvable in polynomial time.

Let $X = \{x_1, x_2, \ldots, x_{3q}\}$ and $C = \{c_1, c_2, \ldots, c_m\}$ with $m \geq q$ be an instance of X3C. Without any loss of generality, we assume that there is a linear order for x's in X, i.e., $x_1 \prec x_2 \prec \cdots \prec x_{3q}$. Let X_2 and X_3 denote the sets of elements of X that are contained in two and three subsets of C, respectively. We assume that $X = X_2 \cup X_3$. From X and C, we construct an undirected graph $G = (V, E)$ as follows.

1. For each $c_i \in C$, we construct a graph $G_c^i = (V_c^i, E_c^i)$ where $V_c^i = \{s_0^i, s_1^i, s_2^i, s_3^i, s_4^i, s_5^i\}$ and $E_c^i = \{(s_0^i, s_3^i), (s_0^i, s_4^i), (s_0^i, s_5^i), (s_1^i, s_2^i), (s_1^i, s_3^i), (s_1^i, s_4^i), (s_2^i, s_3^i), (s_3^i, s_4^i), (s_4^i, s_5^i)\}$. See Fig. 2.2(a).
2. For each $x_i \in X$, we construct a graph $G_x^i = (V_x^i, E_x^i)$. For each $x_i \in X_2$, $V_x^i = \{u_{i-1}, v_{i-1}, x_i\}$ and $E_x^i = \{(u_{i-1}, x_i), (v_{i-1}, x_i)\}$. See Fig. 2.3(a). For each $x_j \in X_3$, $V_x^j = \{w_{j-1}, z_{j-1}, u_{j-1}, v_{j-1}, x_j\}$ and $E_x^j = \{(z_{j-1}, u_{j-1}), (z_{j-1}, v_{j-1}), (u_{j-1}, x_j), (v_{j-1}, x_j), (w_{j-1}, x_j)\}$. See Fig. 2.3(b).
3. We construct graphs $G_x^0 = (V_x^0, E_x^0)$ and $G_x^{3q+1} = (V_x^{3q+1}, E_x^{3q+1})$ where $V_x^0 = \{x_{-1}, x_0\}$, $E_x^0 = \{(x_{-1}, x_0)\}$, $V_x^{3q+1} = \{u_{3q}, v_{3q}, x_{3q+1}, x_{3q+2}\}$, and $E_x^{3q+1} = \{(u_{3q}, x_{3q+1}), (v_{3q}, x_{3q+1}), (x_{3q+1}, x_{3q+2})\}$.
4. We connect vertices in the graphs constructed above by the edges of sets E_{cx} and E_{xx}, where $E_{cx} = \{(x_k, s_0^i) | x_k \in c_i\}$ (see Fig. 2.2(a)) and $E_{xx} = \{(x_i, \alpha_i), (x_i, \beta_i) | i = 0, 1, \ldots, 3q\}$ with $\alpha_i = u_i$ (resp., $\alpha_i = w_i$) and $\beta_i = v_i$ (resp., $\beta_i = z_i$) if $x_{i+1} \in X_2 \cup \{x_{3q+1}\}$ (resp., $x_{i+1} \in X_3$).

The construction of graph $G(V, E)$ is completed by setting

$$V = \bigcup_{i=1}^{m} \{V_c^i\} \cup \bigcup_{i=0}^{3q+1} \{V_x^i\} \quad \text{and} \quad E = \bigcup_{i=1}^{m} \{E_c^i\} \cup \bigcup_{i=0}^{3q+1} \{E_x^i\} \cup E_{cx} \cup E_{xx}.$$

We also set $K' = 3 \cdot m + |X_3| + 4$.

Note that the instance $G = (V, E)$ and K' can be built in polynomial time. We show that the following two *statements* are equivalent.

1. There exists a subcollection $C' \subset C$ for problem X3C.
2. There exists a one-to-one function $f : V \to \{1, \ldots, |V|\}$ such that $\vec{G}_f$ has a total DD-value $\leq K'$.

Given an undirected graph $H = (U, F)$, let $N(u) = \{w | (w, u) \in F\}$ and $d(u) = |N(u)|$ for each $u \in U$. We call $d(u)$ the *degree* of vertex u. Given a linear layout f of H, a vertex $u \in U$ is said to be *balanced* in f if $DD(u) = 1$ and $d(u)$ is odd or $DD(u) = 0$ and $d(u)$ is even. Otherwise, it is said to be *unbalanced*. For subsets $Y, Z \subset U$ with $Y \cap Z = \emptyset$, we use the notation $Y \lhd Z$ to indicate that the vertices of Y are placed to the left of those of Z in f.

Lemma 2.1. If *Statement 1* holds for X and C, then *Statement 2* holds for G and K'.

Proof. Suppose there exists an exact cover $C' \subset C$ for the instance of X and C. We can easily rename the subsets in C such that $C' = \{c_1, c_2, \ldots, c_j\}$ and $C - C' = \{c_{j+1}, \ldots, c_m\}$. The linear layout f of G corresponding to X and C are determined as follows.

1. The vertices of V are placed as $V_c^1 \lhd \cdots \lhd V_c^j \lhd \{x_{-1}\} \lhd \{x_0\} \lhd \{w_0, z_0, u_0, v_0\} \lhd \{x_1\} \lhd \cdots \lhd \{w_{i-1}, z_{i-1}, u_{i-1}, v_{i-1}\} \lhd \{x_i\} \lhd \cdots \lhd \{x_{3q}\} \lhd \{u_{3q}, v_{3q}\} \lhd \{x_{3q+1}\} \lhd \{x_{3q+2}\} \lhd V_c^{j+1} \lhd \cdots \lhd V_c^m$.
2. The vertices of each V_c^i for $i = 1, \ldots, j$ are placed as shown in Fig. 2.2(b). For $i = j+1, \ldots, m$, vertices of each V_c^i are placed in the order opposite to the one shown in Fig. 2.2(b).

A linear layout for the example of Fig. 2.1 thus constructed is shown in Fig. 2.4.

It can be seen that the corresponding $\vec{G}_f$ has the following properties: (1) all even-degree vertices are balanced, (2) all odd-degree vertices except s_1^i of each V_c^i are balanced, and (3) $\mathrm{DD}(s_1^i) = 3$ for each $s_1^i \in V_c^i$. Hence, the total DD-value of $\vec{G}_f$ becomes $3m + |X_3| + 4 = K'$ where the second term is contributed by the $|X_3|$ vertices labeled z that precede $x \in X_3$ and the constant 4 by vertices x_{-1}, x_0, x_{3q+1} and x_{3q+2}. □

Assume that there is an imaginary vertical line going through each vertex $u \in U$ in a linear layout f of $H(U, F)$. We denote the half-plane to the left of the vertical line through vertex u in f as $L[u]$ and the right half-plane as $R[u]$. Let $V_f^L(u)$ (resp., $V_f^R(u)$) be the set of vertices placed in $L[u]$ (resp., $R[u]$) in f. Two linear layouts f and f' of H are said to be *equivalent* if for any $u \in U$, $V_f^L(u) \cap N(u) = V_{f'}^L(u) \cap N(u)$ and $V_f^R(u) \cap N(u) = V_{f'}^R(u) \cap N(u)$, or the vertex order in f is opposite to that in f'. Equivalent layouts have the same total DD-value.

To prove the other direction, we need the following two lemmas.

Lemma 2.2. For any linear layout f of G whose $\vec{G}_f$ has a total DD-value $\le K'$, if vertices $s_1^i, \cdots, s_5^i$ of G_c^i's are placed in $L[s_0^i]$ (resp., $R[s_0^i]$), then vertices x_r, x_s, and x_t in graphs G_x^r, G_x^s, and G_x^t, respectively have to be placed in $R[s_0^i]$ (resp., $L[s_0^i]$).

Note that the sublayout shown in Fig. 2.2(b) satisfies the conditions of Lemma 2.2.

Lemma 2.3. In a linear layout f of G whose $\vec{G}_f$ has a total DD-value $\le K'$, the sublayout for the vertices in $\bigcup_{i=0}^{3q+1}\{V_x^i\}$ is equivalent to the corresponding sublayout constructed in the proof of Lemma 2.1.

Consider the graph constructed for the X3C instance shown in Fig. 2.1. If it has a linear layout with total DD-value ≤ 21, its sublayout for the vertices in $\bigcup_{i=0}^{3q+1}\{V_x^i\}$ must be equivalent to the one shown in Fig. 2.4.

Using the above lemmas, we can prove the reverse of Lemma 2.1 in the following.

Lemma 2.4. If *Statement 2* holds for G and K', then *Satement 1* holds for X and C.

Proof. From Lemma 2.2, we can see that any layout f of G with total DD-value $\le K'$ is equivalent to a layout in which vertices in some V_c^i's are placed to the left of the vertices in $\bigcup_{i=0}^{3q+1}\{V_x^i\}$ and the vertices in the remaining V_c^i's to their right.

Note that every x_i in such layouts must be balanced since by Lemma 2.3 the sublayout corresponding to the vertices in $\bigcup_{i=0}^{3q+1}\{V_x^i\}$ must be equivalent to the one constructed in the proof of Lemma 2.1. Otherwise, the total DD-value of f will be $> K'$. This requires that there be exactly one edge that connect some vertex labeled s_0 in $L[x_i]$ to x_i for each $i = 1, \ldots, 3q$. Hence, those graph G_c^i's whose V_c^i's are placed in $L[x_1]$ in the equivalent layout mentioned above corresponds to a subcollection $C' \subset C$ which exactly covers X. □

It is known that a restricted version of X3C, call *Planar X3C*, is NP-complete [1]. Using an instance of this problem, the construction procedure described above creates a planar graph. The above lemmas remain true in this case. Furthermore, by simply inserting a new vertex into each edge of a planar graph, we form a bipartite planar graph. From this simple transformation, we can obtain the following result.

Theorem 2.1. Problem MTD is NP-complete when G is a bipartite planar graph.

3 Polynomially Solvable Cases

We start with the following theorem which can easily be proved.

Theorem 3.1. The optimization version of probelm MTD is solvable in $O(|V|)$ time when G is a tree or a grid graph.

We now consider the case of a fixed integer K'. Using a modification of Simonson's algorithm [12], we show that MTD is solvable in $O(|V|^{K'/2})$ time.

Let $G(V, E)$ be an undirected graph. For a subset V' of V, a *partial layout* defined on V' is a 1-1 mapping $f_p : V' \to \{1, 2, \ldots, |V'|\}$. Let $DD(f_p) = \sum_{v \in V'} DD(v)$ and $DD_m(f_p) = \sum_{v \in V', d_o(v) < d_i(v)} \{d_o(v) - d_i(v)\}$. Note that $|DD_m(f_p)|$ gives the number of 'completed' arc-disjoint paths contained in f_p. For a vertex $v \in V - V'$, let $c = |N(v) \cap V'|$. The *residue* of the vertex v is defined as:

$$r(v|f_p) = \begin{cases} d(v) \bmod 2 & \text{if } c \le \lceil \frac{d(v)}{2} \rceil, \\ 2c - d(v) & \text{otherwise.} \end{cases}$$

A *representation* of a partial layout f_p is a triplet $\langle M, D, B \rangle$, where $M = \{u \in V' | (u, v) \in E, v \in V - V'\}$, $D = \{(u, v) \in E | u \in M, v \in V - V'\}$, and $B = DD(f_p) + \sum_{v \in V - V'} r(v|f_p)$. The partial layout f_p is said to be *plausible* if it satisfies the following conditions: (1) $|D| \le K'/2 + DD_m(f_p)$ and (2) $B \le K'$.

The following lemma gurantees that if G has a linear layout f with total DD-value $\le K'$, it can always be found by extending its partial layouts.

Lemma 3.1. If two plausible partial layouts f_p^1 and f_p^2 have the same representation, then (1) the sets of vertices in f_p^1 and in f_p^2 are the same, and (2) either both f_p^1 and f_p^2 can be extended to a layout f of G with total DD-value $\le K'$ or neither can.

The above lemma implies that a solution to problem MTD may be found by examining all the plausible partial layouts that have different representations. An algorithm employing this idea is described next.

Let Q be an FIFO queue for storing plausible partial layouts and their representations and P a Boolean array indicating whether a representation $f_p\langle M, D, B \rangle$ is in Q or not.

Algorithm PATH_COVER

Input: An undirected graph $G(V, E)$ and an even integer K'.

Output: A status value S indicating the success or failure of the algorithm.

```
/* $P$ is indexed by the representation of $f_p$, indicated as $P[f_p]$ */
/* $V(f_p)$ represents the set of vertices in $f_p$ */
/* $f_0$ is an 'empty' partial layout */
/* $Q \leftarrow f$ means appending $f$ at the end of $Q$ */
/* $f \leftarrow Q$ means removing the first element from $Q$ and placing it into $f$ */
```

```
begin
  $S \leftarrow$ "FAIL";
  /* Preprocessing */
  if $G$ has more than $K'$ odd-degree vertices or there is a vertex $v \in V$ with $d(v) > K'$
  then output $S$ and stop;
  $Q \leftarrow f_0$; Initialize every entry of $P[f_p]$ to 0;
  while $Q \neq \emptyset$ do
      $f_p \leftarrow Q$; $U \leftarrow V - V(f_p)$;
      while $U \neq \emptyset$ do
          Select a vertex $v \in U$; $U \leftarrow U - \{v\}$;
          /* check plausibility condition (1) */
          if $K'/2 + DD_m(f_p \lhd \{v\}) \geq 0$ and
              $K'/2 \geq |D| - |N(v) \cap V(f_p)| + |N(v) \cap U|$ then
              $f'_p \leftarrow f_p \lhd \{v\}$; /* append $v$ as the last vertex of $f'_p$ */
              Update $f_p\langle M, D, B\rangle$ to $f'_p\langle M', D', B'\rangle$;
              /* check plausibility condition (2) */
              if $D' = \emptyset$ then $S \leftarrow$ "SUCCEED"; Output $S$ and $f'_p$ and stop;
              else if $B' \leq K'$ and $P[f'_p] = 0$ then $Q \leftarrow f'_p$; $P[f'_p] \leftarrow 1$;
      end while;
  end while;
  Output $S$;
end
```

Note that $|D| - |N(v) \cap V(f_p)| + |N(v) \cap U|$ is the number of edges that connect the vertices in $V(f_p) \cup \{v\}$ and those in U and should be $\leq K'/2$ if $V(f_p) \lhd \{v\}$ is a feasible layout.

From Lemma 3.1, we can see that plausible partial layouts with different representations need to be inserted into Q exactly once. Algorithm PATH_COVER accomplishes this by using the array P. Note that the time spent on algorithm PATH_COVER is proportional to the number of different representations pushed into Q. Since vertex set M can have at most $K'/2$ elements with a uniquely determined D for each of them and $O(B) = O(1)$, algorithm PATH_COVER has time complexity $O(|V|^{K'/2})$. Thus we have the following theorem.

Theorem 3.2. Probelm MTD is solvable in $O(|V|^{K'/2})$ time when K' is given.

4 A Heuristic Algorithm

We now describe a greedy heuristic algorithm for general graphs. This is done by first forming a *trivial* partial layout consisting of a single vertex, then extending it one vertex at a time until a linear layout with a minimal total DD-value is obtained. An estimate of the total DD-value is calculated for each partial layout formed during layout extensions. Whenever the estimate exceeds a precomputed upper bound, the corresponding partial layout is discarded.

Let $\{u_1\} \lhd \{u_2\} \lhd \cdots \lhd \{u_n\}$ be a layout of $G(V, E)$. Let $f_{i,j}$ denote a partial layout corresponding to $\{u_i\} \lhd \{u_{i+1}\} \lhd \cdots \lhd \{u_j\}$ and $DD(f_{1,i}) = \sum_{k=1}^{i} DD(u_k)$. Let $DD(v|f_{1,i})$ be the DD-value of vertex v when v is placed immediately after the partial layout $f_{1,i}$ and $DD(f_{i+1,j}|f_{1,i}) = \sum_{k=i+1}^{j} DD(v_k|f_{1,i} \lhd \{v_{i+1}\} \lhd \cdots \lhd \{v_{k-1}\})$. Finally, we define the *level-k* cost of $v \in V - V(f_p)$, denoted by $C^k(v|f_p)$, as

$$\begin{aligned} C^k(v|f_p) = DD(v|f_p) \;&+ \min_{V(f_p')\subset V-V(f_p)-\{v\}} DD(f_p'|f_p \lhd \{v\}) \\ &+ \; R(f_p \lhd \{v\} \lhd f_p') \end{aligned}$$

where $f_p' \equiv \{v_1\} \lhd \{v_2\} \lhd \cdots \lhd \{v_k\}$ with $\{v_1, v_2, \ldots, v_k\} \subset V - V(f_p) - \{v\}$ and $R(f_p \lhd \{v\} \lhd f_p') = \sum_{u \in V - V(f_p) - \{v\} - V(f_p')} r(u|f_p \lhd \{v\} \lhd f_p')$.

Note that the second term of $C^k(v|f_p)$ calculates the minimum possible increase of DD-value when k additional vertices are placed after v and that the last term provides the residues of the remaining vertices. Hence, $C^k(v|f_p)$ calculates a lower bound on the increase of the total DD-value of a linear layout f which is an extension of the partial layout $f_p \lhd \{v\} \lhd f_p'$. Therefore, the construction of a layout f may be terminated whenever its lower bound on the total DD-value exceeds the total DD-value of a best layout that we have obtained so far. The total DD-value of the currently best completed layout is used as an upper bound for the DD-value estimates of partial layouts, which is initially set to be ∞.

Algorithm GREEDY

Input: An undirected graph $G(V, E)$ and a nonnegative integer k specifying the 'cost level'.

Output: A linear layout f of G with minimal total DD-value.

```
/* $f$ – a currently best layout of $G$, $f_{1,0}$ – the intial 'empty' partial layout */
begin
    $V_0 \leftarrow V$;
    while $V_0 \neq \emptyset$ do    /* Outer loop */
        $U \leftarrow V_0$; $DD(f) \leftarrow \infty$; $DD(f_{1,0}) \leftarrow 0$; $i \leftarrow 1$;
        while $U \neq \emptyset$ do    /* Inner loop */
            /* Order vertices in $U$ if $i > 1$ – Method 2 */
            Select vertex $v \in U$ such that $C^k(v|f_{1,i-1}) = \min_{u \in U} C^k(u|f_{1,i-1})$;
            if $DD(f_{1,i-1}) + C^k(v|f_{1,i-1}) < DD(f)$ then
                Place $v$ as the $i$th vertex of $f_{1,i}$;
                $DD(f_{1,i}) \leftarrow DD(f_{1,i-1}) + DD(v|f_{1,i-1})$;
                Update cost $C^k(u|f_{1,i})$ for each $u \in V - V(f_{1,i})$;
                if $i = 1$ then $U \leftarrow V - \{v\}$; else $U \leftarrow U - \{v\}$;
                $i \leftarrow i + 1$;
            else break;    /* current partial layout is discarded */
        end while;
        /* a new layout is completed */
        if $U = \emptyset$ and $DD(f_{1,n}) < DD(f)$ then $f \leftarrow f_{1,n}$; $DD(f) \leftarrow DD(f_{1,n})$;
        $V_0 \leftarrow V_0 - \{v\}$;
    end while;
    Output $f$; /* Current $f$ is a best layout found by GREEDY */
end
```

Algorithm GREEDY terminates after all the trivial partial layouts and their extensions are considered. Hence, the time complexity of algorithm GREEDY is $O(|V| \cdot T)$ where T is the time spent on the iterations of the inner while loop. Note that T is dominated by the time spent on updating the costs of the vertices in $V - V(f_{1,i})$, which is $O(\sum_{x=1}^{|V|} x^{k+1}) = O(|V|^{k+2})$. Hence, algorithm GREEDY has time complexity $O(|V|^{k+3})$ when cost level k is specified.

Algorithm GREEDY has been implemented in C and tested on a SPARC workstation for some randomly generated graphs of 32 and 64 nodes. These experimental results are listed in Tables 1 and 2. The difference between Methods 1 and 2 listed in the tables lies in the 'vertex selection' step. Method 1 scans the vertices of U in an arbitrary order. Method 2 first sorts the vertices in such a way that those having neighbors in the current partial layout will be considered first. Each entry in the tables consists of the total DD-value of the layout created when different cost levels are specified and its associated running time in seconds (in parentheses).

From Tables 1 and 2 we can see that using a cost function with a higher level does not necessarily produces better solutions but consumes a hugh amount of time. It seems that $k = 0$ or 1 is sufficient for all the cases we show. In fact, as the average degrees of the graphs increase, the size of a path cover created by GREEDY approaches that of the maximum cut of the corresponding linear layout. Further analysis of the performance of GREEDY is currently under way.

References

1. M. E. Dyer and A. M. Frieze, "Planar 3DM is NP-Complete," *J. of Algorithms,* Vol. 7, pp. 174-184, 1986.
2. G. N. Frederickson and S. E. Hambrusch, "Planar Linear Arrangements of Outerplanar Graphs," *IEEE Trans. on Circuits and Systems*, Vol. 35, No. 3, pp. 323-333, March 1988.
3. P. Gee, M. Y. Wu, S. M. Kang, and I. N. Hajj, "Metal-Metal Matrix (M^3) CMOS Cell Generator with Compaction," in *Proc. Int'l Conf. on Computer-Aided-Design,* Santa Clara, CA., Nov. 1987, pp. 184-187.
4. M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Thoery of NP-Completeness*, San Francisco: Freeman, 1979.
5. M. Garey, D. Johnson, and L. Stockmeyer, "Some Simplified NP-Compl ete Graph Problems," *Theor. Comput. Sci.* Vol. 1, pp. 237-267, 1976.
6. S. M. Kang, "Metal-Metal Matrix (M^3) for High-speed MOS VLSI Lay out," *IEEE Trans. Computer-Aided-Design,* Vol. CAD-6, No. 5, pp. 886-891, Sept. 1987.
7. D. Kratsch, "Finding the Minimum Bandwidth of an Interval Graph," *Information and Computation*, Vol. 74, pp. 140-158, 1987.
8. J. Y-T. Leung, O. Vornberger, and J. D. Witthoff, "On Some Variants of the Bandwidth Minimization Problem," *SIAM J. Comput.*, Vol. 13, No. 3, pp. 650-667, August 1984.
9. F. S. Makedon, C. H. Papadimitriou, and I. H. Sudborough, " Topological Bandwidth," *SIAM J. Algebraic Discrete Methods*, Vol. 6, No. 3, pp. 418-444, July 1985.
10. B. Monien, "The Bandwidth Minimization for Caterpillars with Hair Length 3 is NP-Complete," *SIAM J. Algebraic Discrete Methods*, Vol. 7 , No. 4, pp. 505-512, Oct. 1986.
11. Y. Shiloach, "A Minimum Linear Arrangement Algorithm for Undirected Trees," *SIAM J. Comput.*, Vol. 8, No. 1, pp. 15-32, Feb. 1979.
12. S. Simonson, "Routing with Critical Paths," *Information Processing Letters,* Vol. 34, pp. 174-184, Feb. 1990.
13. M. Yannakakis, "A Polynomial Algorithm for the Min-cut Linear Arrangement of Trees," *J. ACM,* Vol. 32, No. 4, pp. 950-988, Oct. 1985.

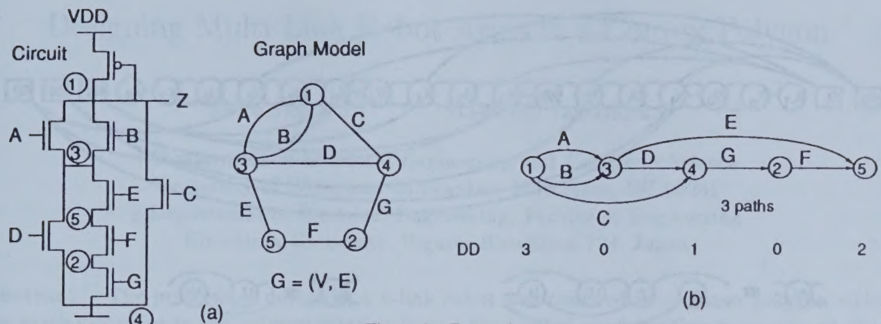
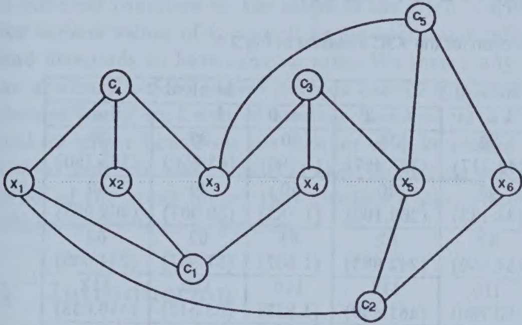

VDD
Circuit
Graph Model
1
Z
A
B
3
E
C
5
D
F
2
G
VSS
4
(a)
G = (V, E)
A
B
C
D
E
F
G
3 paths
DD 3 0 1 0 2
(b)

Fig.1. An Example.


c_5
c_4
c_3
x_1
x_2
x_3
x_4
x_5
x_6
c_1
c_2

$X = \{x_1, x_2, x_3, x_4, x_5, x_6\}$

$C = \{ \{x_1, x_2, x_4\},$ c_1 ✓
$\{x_1, x_5, x_6\},$ c_2
$\{x_3, x_4, x_5\},$ c_3
$\{x_1, x_2, x_3\},$ c_4
$\{x_3, x_5, x_6\} \}$ c_5 ✓

$X_2 = \{x_1, x_3, x_5\}$
$X_3 = \{x_2, x_4, x_6\}$
$C' = \{c_1, c_5\}$

Fig.2.1. An instance of X3C.

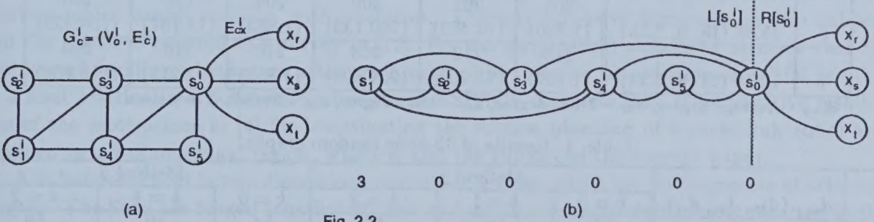

$G_c^l = (V_0^l, E_c^l)$
E_{cx}^l
s_2^l
s_3^l
s_0^l
s_1^l
s_4^l
s_5^l
x_r
x_s
x_t
(a)
$L[s_0^l]$
$R[s_0^l]$
3 0 0 0 0 0
(b)

Fig. 2.2.

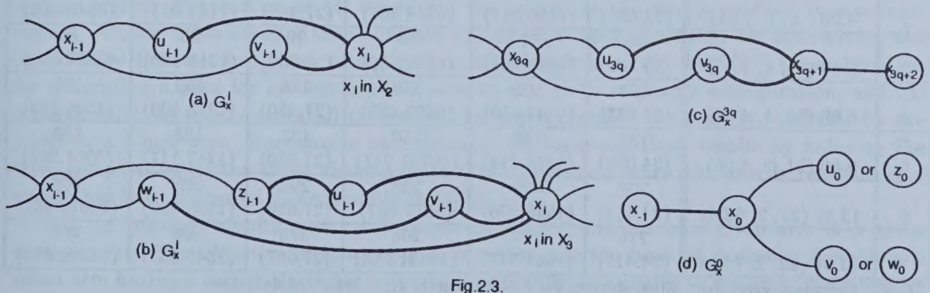
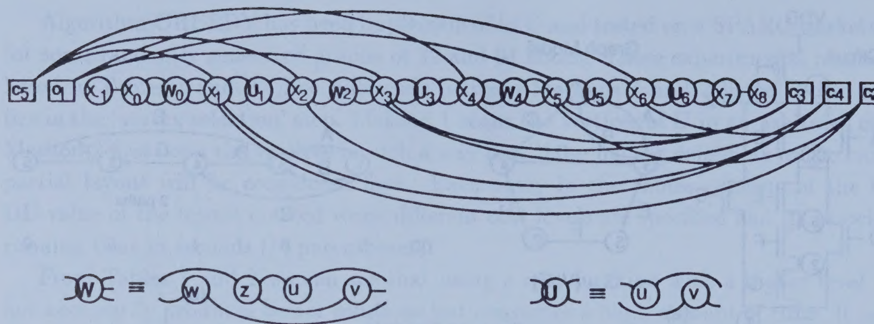

x_{l-1}
u_{l-1}
v_{l-1}
x_l
x_l in X_2
(a) G_x^l
x_{3q}
u_{3q}
v_{3q}
x_{3q+1}
x_{3q+2}
(c) G_x^{3q}
x_{l-1}
w_{l-1}
z_{l-1}
u_{l-1}
v_{l-1}
x_l
x_l in X_3
(b) G_x^l
u_0 or z_0
x_{-1}
x_0
v_0 or w_0
(d) G_x^0

Fig.2.3.



$K' = 3m + |X_3| + 4 = 15 + 3 + 4 = 21$

Fig.2.4. Construction for the X3C instance of Fig.2.1.

No.	d_{avg} (d_M, d_m, d_{st})	Method 1			Method 2		
		$k = 0$	$k = 1$	$k = 2$	$k = 0$	$k = 1$	$k = 2$
1	2.88 (3, 1, 0.48)	42 (1.700)	36 (42.117)	36 (367.467)	40 (1.700)	36 (43.650)	36 (363.150)
2	5.75 (6, 4, 0.61)	40 (1.583)	40 (48.133)	40 (269.100)	40 (1.783)	40 (50.967)	40 (268.983)
3	7.94 (9, 3, 1.69)	64 (1.767)	62 (54.550)	62 (242.983)	64 (1.867)	62 (57.867)	62 (244.500)
4	11.00 (12, 6, 1.62)	110 (1.833)	110 (60.200)	114 (461.450)	110 (1.833)	110 (63.517)	112 (456.033)
5	13.75 (15, 6, 2.57)	174 (1.717)	166 (68.600)	166 (329.200)	174 (1.833)	166 (68.700)	166 (335.917)
6	15.88 (18, 9, 2.55)	204 (1.800)	202 (72.483)	200 (760.133)	204 (1.883)	196 (71.767)	200 (766.333)
7	16.19 (21, 8, 3.24)	210 (1.767)	210 (73.200)	208 (959.800)	210 (1.850)	210 (72.217)	208 (956.950)

d_{avg} – Average degree, d_M – Max. degree, d_m – Min. degree, d_{st} – Standard deviation

Table. 1. Results of 32-node random graphs.

No.	d_{avg} (d_M, d_m, d_{st})	Method 1			Method 2		
		$k = 0$	$k = 1$	$k = 2$	$k = 0$	$k = 1$	$k = 2$
1	2.28 (3, 1, 0.72)	58 (23.633)	50 (1088.533)	52 (12858.117)	54 (26.767)	50 (1095.333)	50 (12845.050)
2	4.00 (6, 1, 1.51)	66 (23.683)	66 (1150.017)	70 (6264.567)	68 (27.367)	68 (1144.933)	70 (6340.000)
3	6.09 (11, 1, 2.87)	90 (23.700)	92 (1225.933)	84 (5121.050)	90 (26.350)	86 (1217.150)	86 (5857.500)
4	8.88 (16, 1, 4.08)	128 (23.683)	132 (1342.550)	132 (9982.483)	128 (27.450)	132 (1323.400)	128 (8509.233)
5	9.00 (21, 2, 4.98)	132 (24.050)	128 (1374.417)	130 (10797.917)	132 (27.550)	128 (1342.517)	130 (9961.967)
6	13.56 (27, 2, 6.94)	290 (24.433)	300 (1536.600)	288 (9677.667)	290 (27.650)	296 (1526.400)	288 (10678.467)
7	13.97 (29, 2, 7.21)	276 (24.417)	268 (1551.317)	268 (11421.183)	276 (27.567)	268 (1540.633)	268 (11345.067)

d_{avg} – Average degree, d_M – Max. degree, d_m – Min. degree, d_{st} – Standard deviation

Table. 2. Results of 64-node random graphs.

Designing Multi-Link Robot Arms in a Convex Polygon

Ichiro Suzuki † Masafumi Yamashita ‡

† Department of Electrical Engineering and Computer Science
University of Wisconsin–Milwaukee, Milwaukee, WI 53201
‡ Department of Electrical Engineering, Faculty of Engineering
Hiroshima University, Higashi-Hiroshima 724, Japan

Abstract The problem of designing a k-link robot arm confined in a convex polygon which can reach any point in the polygon starting from a fixed initial configuration is considered. The links of an arm are assumed to be all of the same length. We present a necessary condition and a sufficient condition on the shape of the given polygon for the existence of such a k-link arm for various values of k, as well as necessary and sufficient conditions for rectangles, triangles and diamonds to have such an arm. We then study the case $k = 2$, and show that whether an arbitrary n-sided convex polygon has a 2-link arm which can reach any inside point can be decided and if so, a desired arm can be found in $O(n^2)$ time. Finally, we prove a lower bound and an upper bound on the number of links needed to construct such an arm for a general convex polygon, and show that the two bounds can differ by at most one. The constructive proof of the upper bound thus provides a simple method for designing a desired arm having almost as few links as possible.

1 Introduction

Motion planning is one of the most fundamental issues in robotics and automation. The problem is to plan a path of a given object (either solid or hinged) from an initial position to a final position which avoids collision with obstacles. There is extensive literature on this subject, and the reader is referred to a survey in [12]. For our purposes, it suffices to mention that the problem of deciding whether an arbitrary hinged object can be moved from an initial position to a final position in a 3-dimensional region is PSPACE-complete [11]. This negative result is one of the motivations in [4] for investigating the motion planning of a multi-link robot arm confined in a 2-dimensional region, which is also the subject of the current paper.

A *k-link robot arm* in two dimensions consists of k *links*, which are line segments of arbitrary lengths. The links are hinged together at their endpoints consecutively by freely rotating joints. One endpoint of the arm, called the *shoulder*, is pinned down, but the link having the shoulder as one endpoint can rotate freely about the shoulder. The entire arm is restricted to stay within a given region. Hopcroft *et al.* [4] considered the case when the arm is confined in a circular region. Among other results, they obtained (1) an $O(k)$ time algorithm for determining the reachability between two given configurations of a k-link arm, (2) an $O(k^3)$ time algorithm for generating a plan for moving a k-link arm to any given reachable configuration, and (3) a polynomial (at least quadratic) time algorithm for computing the regions reachable by the joints of a k-link arm. Kantabutra and Kosaraju [7] improved these results by reducing the time complexities of the three algorithms to $O(k)$. The case when the arm is confined in a square has been discussed more recently in [5][6].

All the previous results mentioned above are on the *analysis* of a given arm in a given environment. The objective of this paper is to investigate the issue of *designing* a multi-link robot arm having a certain desirable property. Specifically, our goal is to design, given a convex polygon P, a k-link arm confined in P such that (1) (*manageability*) the tip of the arm can be moved to any point in P starting from a fixed initial configuration, and (2) (*minimality*) the number k of links is as small as possible. An arm satisfying the first condition is said to *manage* P. A convex polygon is said to be *k-manageable* if there exists a k-link arm which manages it. Manageability can be a very desirable property in certain applications, since it allows an arm to pick up and drop an object anywhere in the given (convex) workspace. Note that a 2-link arm can manage a polygon only if the lengths of the two links are equal, since otherwise the tip of the arm cannot reach the shoulder. Also, it is conceivable that in practice, we may wish to build an arm for a given particular application using only "standard" links of some fixed length. For this reason, we only consider the case in which all links of an arm must be of the same length.

The main results of this paper are the following. First, we present a simple sufficient condition for a general convex polygon P to be k-manageable for various values of k, using the concept of *roundness* (the maximum ratio of the radii of two concentric circles which enclose the boundary of P between them). Next, we give a necessary condition for P to be k-manageable for various values of k in terms of the smallest internal angle at the corners of P, and use it to prove simple necessary and sufficient conditions for rectangles, triangles and diamonds (i.e., parallelograms whose sides are all equal) to be k-manageable. The proofs of the sufficiency are constructive, providing simple methods for designing a desired arm for these elementary polygons under the given conditions. We then study the case $k = 2$, and show that whether an arbitrary n-sided convex polygon is 2-manageable can be decided and if so, a desired 2-link arm can be found in $O(n^2)$ time. Finally, we prove a lower bound and an upper bound on the number of links needed to manage a general convex polygon P, and show that they can differ by at most one. The constructive proof of the upper bound thus provides a simple method for designing an arm managing P having almost as few links as possible. The two bounds are given using a parameter related to the roundness of P.

Necessary definitions are given in Section 2. The results related to roundness are presented in Section 3. Section 4 and Section 5 discuss the results on elementary polygons and 2-manageability, respectively. Upper and lower bounds on the number of links needed to manage a general convex polygon are proved in Section 6. Concluding remarks are found in Section 7. Due to space limitation, we present only the key ideas and some proof outlines. All the missing details can be found in the full paper.

2 Definitions

A *k-link robot arm*, $k \geq 2$, is a sequence of k line segments $L_1, \ldots, L_k$ called *links* which are hinged together at their endpoints by freely rotating joints. The links are all of the same length. The endpoints of the links are labeled $A_0, \ldots, A_k$, where A_0, which is fixed in the plane, and A_k are called the *shoulder* and *tip*, respectively, and A_i, $1 \leq i \leq k-1$, is the joint connecting L_i and L_{i+1}. The *joint angle* at A_i, $1 \leq i \leq k-1$, is the angle between L_i and L_{i+1}. The joint angle at A_0 is the angle between L_1 and some reference line through A_0. See Figure 1.

Throughout the paper we assume that the entire arm is constrained to be inside a given convex polygon P at all times. Two configurations (or placements) of the arm are mutually *reachable* if it is possible to move the arm from one configuration to the other by allowing the joint angles to change simultaneously (in such a way that the arm lies within P at all times, of course). A point $x \in P$ is said to be *reachable* from a configuration c if there exists another configuration c' reachable from c such that the tip A_k is at x in c'. We say that the arm *manages* P if there exists a configuration from which all the points in P are reachable, i.e., the tip of the arm can be moved to any point in P starting from some initial configuration. P is said to be *k-manageable* if there exists a k-link arm which manages P.

Example 1 Figure 2 shows a 3-manageable polygon and two configurations of a 3-link arm which manages it. Now we show that the polygon is not 2-manageable. Note that the length of the links of any 2-link arm managing the polygon must be at least 1 since the distance between

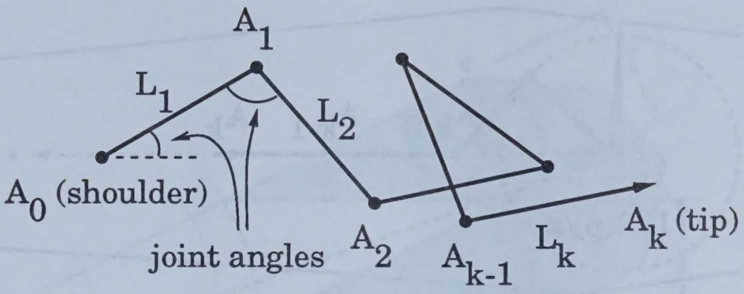


Figure 1: A k-link robot arm.

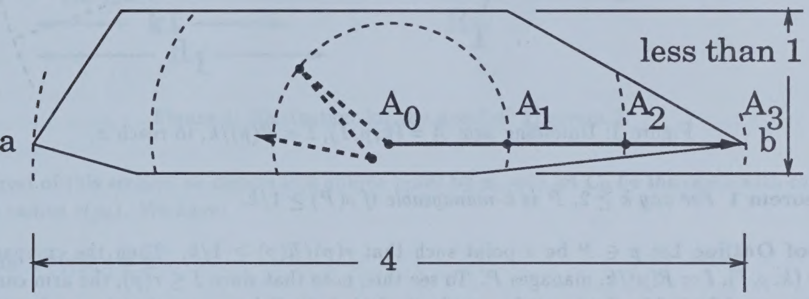


Figure 2: A 3-link arm managing a convex polygon.

corners a and b is 4. However, the "height" of the polygon (as indicated in the figure) is less than 1, and hence the links of such an arm would not be "reorientable," i.e., if the tip A_2 is to the left of joint A_1 initially, then it remains to be so regardless of the movement. Therefore the tip cannot be moved to at least one of a and b.

We use the following notation. For points x and y, $|xy|$ denotes the distance between x and y. The boundary of a polygon P is denoted by ∂P. For a point $p \in P$, $r(p)$ denotes the distance between p and ∂P, and $R(p)$ denotes the distance from p to the furthest corners of P. For a corner v of P, $\angle v$ denotes the inner angle at corner v. We denote by $\mathcal{A} = (k, p, I)$ the k-link arm $\mathcal{A}$ whose shoulder is at point p and whose link length is I. Arm $\mathcal{A} = (k, p, I)$ is said to be *compact* if $I = R(p)/k$, i.e., the total length of the arm is equal to the distance from the shoulder to the furthest corners of P. The 3-link arm shown in Figure 2 is compact.

3 Roundness of a Convex Polygon

Intuitively, convex polygons having a rounder profile should require fewer links to manage than those having a more out-of-round profile. To study the relation between the profile of a polygon and the number of links needed to manage it, we define the *roundness* of a convex polygon P by $\rho(P) = \max_{p \in P} r(p)/R(p)$. That is, $\rho(P)$ is the maximum of the ratio of the radii of two concentric circles which enclose the boundary of P between them. For example, the roundness of a rectangle with dimensions $1 \times x$ approaches zero as x diverges from 1, and the roundness of a regular n-gon approaches one as n grows larger. (There are other measures of roundness. One of them is $\min_{p \in P}(R(p) - r(p))$, called the *minimum radial separation* [8]. A similar measure for a point set is considered in [3].) The following theorem gives a sufficient condition for P to be k-manageable.

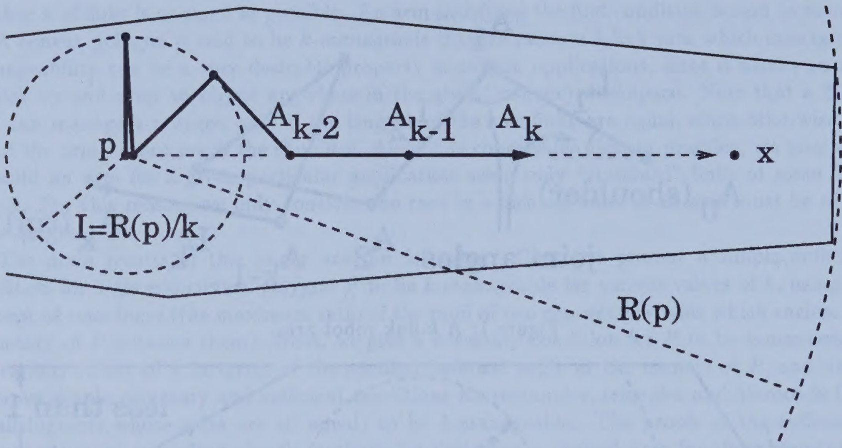


Figure 3: Unfolding arm $\mathcal{A} = (k, p, I)$, $I = R(p)/k$, to reach x.

Theorem 1 *For any $k \geq 2$, P is k-manageable if $\rho(P) \geq 1/k$.*

Proof Outline Let $p \in P$ be a point such that $r(p)/R(p) \geq 1/k$. Then the compact arm $\mathcal{A} = (k, p, I)$, $I = R(p)/k$, manages P. To see this, note that since $I \leq r(p)$, the arm can make a rotation of 2π inside P when it is completely folded. So if k is even, then to reach any given point $x (\neq p) \in P$ we first rotate the entire arm (without unfolding it) about p until the links are perpendicular to $\overline{px}$, and then pull the tip A_k straight toward x (Figure 3). Clearly, none of the joints can be blocked by the boundary of P during this process since P is convex, and the tip A_k eventually reaches x since $|px| \leq kI$. If k is odd and $|px| \geq I$, then we must rotate link L_k for $\pi/2$ and place A_k on $\overline{px}$ before we pull A_k toward x. A slightly different operation is used if k is odd and $|px| < I$. $\square$

The condition $\rho(P) \geq 1/k$ is not necessary for P to be k-manageable. In fact, we have already seen that the polygon shown in Figure 2 is 3-manageable, but it is easy to verify that its roundness is less than $1/3$. The key observation here is that the link L_1 need not be able to make a complete rotation of 2π inside P for the arm to manage P. We will return to this subject and present a stronger sufficient condition in Section 6.

In the rest of this section we give some useful facts regarding roundness, and present a method for computing the value of $\rho(P)$. Let $M(P)$ be the set of points p in the interior of P such that there are at least two points on the boundary of P that are equidistant from p and are closest to p. $M(P)$ is called the *medial axis* of P [9]. Let $V(P)$ be the set of points p such that there are at least two corners of P that are equidistant from p and are furthest from p. $V(P)$ is the *furthest-neighbor Voronoi diagram* [10] for the set of corners of P. In the following, $\gamma(p)$ denotes the set of corners of P which are furthest from a point $p \in P$. The following lemma is easy to prove.

Lemma 1 *If $\rho(P) = r(p)/R(p)$, then $p \in M(P)$.*

In general, there may exist more than one point $p \in P$ such that $\rho(P) = r(p)/R(p)$. However, since P is convex, there is a unique point p such that (1) $\rho(P) = r(p)/R(p)$ and (2) $R(p) \leq R(q)$ for all q such that $\rho(P) = r(q)/R(q)$. (To see this, suppose that both p and $p' (\neq p)$ have these properties. Then since $R(p) = R(p')$, $r(p) = r(p')$ and P is convex, there must exist a point $q \in \overline{pp'}$ such that $R(q) < R(p)$ and $r(q) \geq r(p)$. Thus $r(q)/R(q) > r(p)/R(p)$, a contradiction.)

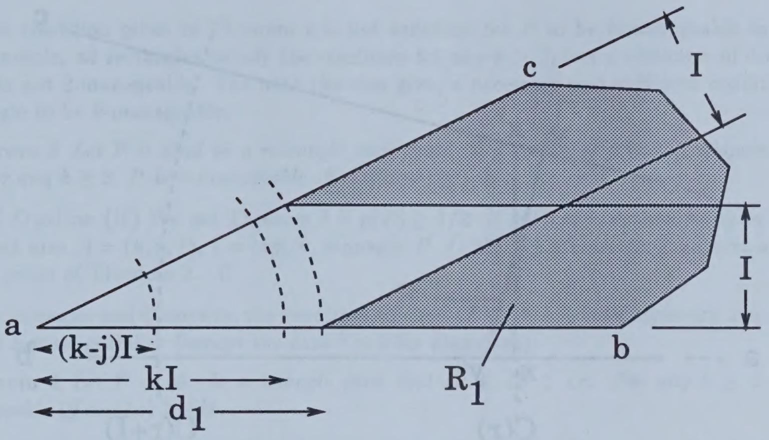


Figure 4: Illustration for the proof of Theorem 2.

In the rest of this section we denote this unique point by p_0, and let C_0 be the circle with center p_0 and radius $r(p_0)$. We have:

Lemma 2 $|\gamma(p_0)| \geq 2$.

Lemma 2 implies that $p_0 \in V(P)$. By this and Lemma 1, $p_0 \in M(P) \cap V(P)$. If P has n vertices, then both $M(P)$ and $V(P)$ can be computed in $O(n)$ time [1], and since each of $M(P)$ and $V(P)$ contains $O(n)$ edges, the intersection $M(P) \cap V(P)$ can be computed in $O(n \log n + m)$ time where m is the total number of intersections [2]. Thus we can find p_0 and compute $\rho(P)$ in $O(n \log n + m)$ time. (The method for computing $\rho(P)$ described above is essentially the same as that given in [8] for computing the minimum radial separation. In fact, we can show that $R(p_0) - r(p_0)$ is the minimum radial separation of P.)

4 k-Manageability of Elementary Polygons

In this section we present simple necessary and sufficient conditions for rectangles, triangles and diamonds to be k-manageable. Our results show that finding an arm managing such elementary polygons is straightforward.

The following theorem gives a necessary condition for a general convex polygon P to be k-manageable. We denote by $\min \angle P$ the smallest inner angle at the corners of P.

Theorem 2 *For any $k \geq 2$, if P is k-manageable, then either* $\min \angle P > \pi/2$ *or* $\sin(\min \angle P) \geq 1/k$.

Proof Let a be a corner of P such that $\angle a \leq \pi/2$ and $\sin(\angle a) < 1/k$. Suppose that there exists a k-link arm $\mathcal{A} = (k, p, I)$ which manages P. Let R_1 be the region consisting of those points of P which are at distance I or larger from either the line containing edge ab or the line containing edge ac, where b and c are the corners of P adjacent to a (Figure 4). For $0 \leq i \leq k-1$, link L_{i+1} is said to be *reorientable* if there exists a reachable configuration of $\mathcal{A}$ (i.e., a configuration reachable from the initial configuration) in which joint A_i is in R_1. Clearly L_k must be reorientable, since otherwise the tip A_k cannot be moved to both a and the points in P which are furthest from a. (This shows that R_1 is not empty.) If we let d_1 be the distance between a and R_1, then $d_1 \geq I/\sin(\angle a) > kI$. From this and $|ap| \leq kI$ (i.e.,

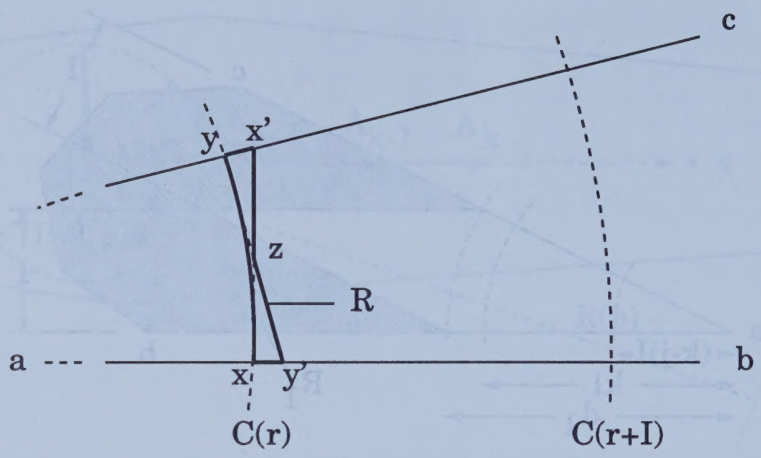


Figure 5: The region where A_0 lies.

the arm is long enough to reach a), we have $p \notin R_1$, i.e., L_1 is not reorientable. So let j be the smallest index such that link L_{j+1} is reorientable, and define two other regions R_2 and R_3; R_2 is the set of points $q \in P$ such that $|aq| \leq (k-j)I$, and R_3 is the set of points $q \in P$ such that $|aq| \geq kI$. Then A_j must be able to reach both R_2 (since otherwise the total length $(k-j)I$ of links $L_{j+1}, \ldots, L_k$ is not large enough to reach a) and R_3 (since A_j can reach R_1 and $R_1 \subseteq R_3$ by $d_1 > kI$). Now we show that this is not possible. Specifically, we prove the following proposition.

Proposition 1 *Let $C(\ell)$ be the perimeter of the disc having center a and radius ℓ. For any $1 \leq i \leq j$ and $0 \leq r \leq (k-i)I$, A_i cannot reach both $C(r)$ and $C(r+iI)$.*

Proof Since L_1 is not reorientable, it can become perpendicular to neither $\overline{ab}$ nor $\overline{ac}$. Therefore if A_1 can reach both $C(r)$ and $C(r+I)$ for some $0 \leq r \leq (k-1)I$, then A_0 must be in region $R = xy'zx'y$ shown in Figure 5 determined by arc $\widehat{xy}$ of $C(r)$ and line segments $\overline{xx'}$ and $\overline{yy'}$ perpendicular to $\overline{ab}$ and $\overline{ac}$, respectively. (z is the intersection of $\overline{xx'}$ and $\overline{yy'}$.) Since $r \leq (k-1)I$ and $\tan(\angle a) = \sin(\angle a)/\sqrt{1-\sin^2(\angle a)} < 1/\sqrt{k^2-1}$, we have $|xx'| = |yy'| = r\tan(\angle a) < ((k-1)/\sqrt{k^2-1})I < I$. Thus the distance between any point in R and any point on arc $\widehat{xy}$ is less than I, and hence A_0 cannot be in R. Since this is a contradiction, we conclude that A_1 cannot reach both $C(r)$ and $C(r+I)$ for any $0 \leq r \leq (k-1)I$. Suppose that for some $2 \leq i \leq j$ and $0 \leq r \leq (k-i)I$, A_i can reach both $C(r)$ and $C(r+iI)$. Since L_i is not reorientable, either

(1) when A_i is on $C(r)$, A_{i-1} is inside the disc with perimeter $C(r)$, or

(2) when A_i is on $C(r+iI)$, A_{i-1} is outside the disc with perimeter $C(r+iI)$.

This implies either

(1′) A_{i-1} can reach both $C(r+I)$ and $C(r+iI)$, or

(2′) A_{i-1} can reach both $C(r)$ and $C(r+(i-1)I)$.

Since $0 \leq r < r+I \leq (k-(i-1))I$, the proposition follows by induction. □

Thus arm $\mathcal{A}$ does not manage P. This is a contradiction. □

The condition given in Theorem 2 is not sufficient for P to be k-manageable in general. For example, all rectangles satisfy the condition for any $k \geq 2$, but a rectangle of dimensions 1×4 is not 2-manageable. The next theorem gives a necessary and sufficient condition for a rectangle to be k-manageable.

Theorem 3 *Let $P = abcd$ be a rectangle such that $|ab| \leq |bc|$. Let p be the midpoint of edge bc. For any $k \geq 2$, P is k-manageable iff* $\sin(\angle apb) \geq 1/k$.

Proof Outline (if) We use Theorem 1 if $\rho(P) \geq 1/2$. If $\rho(P) < 1/2$, then we show that the compact arm $\mathcal{A} = (k, p, I)$, $I = |ap|/k$, manages P. **(only if)** We use an argument analogous to the proof of Theorem 2. □

For triangles and diamonds, the condition $\sin(\min \angle P) \geq 1/k$ is both necessary and sufficient for the k-manageability (except the case $k = 2$ for diamonds).

Theorem 4 *Let $P = abc$ be a triangle such that $\angle a \leq \angle b \leq \angle c$. For any $k \geq 2$, P is k-manageable iff* $\sin(\angle a) \geq 1/k$.

Proof Outline (if) We show that the compact arm $\mathcal{A} = (k, p, I)$ manages P, where p is the intersection of edge ab and the bisector of corner c and $I = \max\{|ap|, |cp|\}$. **(only if)** This part follows from Theorem 2 since $\angle a \leq \pi/3$. □

Theorem 5 *Let $P = abcd$ be a diamond such that $\angle a \leq \pi/2$. For any $k \geq 3$, P is k-manageable iff* $\sin(\angle a) \geq 1/k$.

Proof Outline (if) We use Theorem 1 if $\rho(P) \geq 1/2$. If $\rho(P) < 1/2$, then we show that the compact arm $\mathcal{A} = (k, b, I)$ with shoulder b and $I = |ab|/k$ manages P, using an argument similar to that given in the proof of Theorem 1 with a slight twist to cope with the case when the folded arm cannot become perpendicular to $\overline{bx}$, where x is the given point to be reached. **(only if)** This part follows from Theorem 2 since $\angle a \leq \pi/2$. □

For $k = 2$, the construction given in the proof of Theorem 5 (in which A_0 is placed on b) does not work. To see this, let y and z be the two extreme positions of joint A_1 on edges ab and bc, respectively. Then since $\angle abc$ is less than π, the points $x \in P$ such that $|xy| < I$ and $|xz| < I$ are not reachable. In Section 5, using a number of additional lemmas we show that the diamond P is 2-manageable iff $\tan(\angle a) \geq (1 + \sqrt{3})/2$.

5 2-Manageability

In this section we consider the problem of deciding whether a convex polygon P is 2-manageable and if so, finding a 2-link arm which manages P. The following lemma is easy to prove.

Lemma 3 *Suppose that P is managed by $\mathcal{A} = (2, p, I)$. Then P is managed by the compact arm $\mathcal{A}' = (2, p, R(p)/2)$.*

Let us see how a compact 2-link arm $\mathcal{A} = (2, p, I)$, $I = R(p)/2$, manages a convex polygon P when $\rho(P) < 1/2$. Let v be any one of the corners of P furthest from p, and suppose that the arm is fully stretched and the tip A_2 is at v. Let L_p and $R_p \in \partial P$ be the positions of joint A_1 when link L_1 has been rotated counterclockwise and clockwise, respectively, as much as possible starting from this configuration (ignore link L_2 during this rotation). See Figure 6. The points L_p and R_p, called the *left* and *right stops* of p, respectively, always exist since $\rho(P) < 1/2$. Let α_p be the apex angle of the wedge containing v determined by the line segments $\overline{pL_p}$ and $\overline{pR_p}$. As is seen from the discussion given at the end of Section 4, the arm cannot manage P unless $\alpha_p \geq \pi$. Furthermore, the points on the boundary of P between L_p and R_p must be within distance I from either L_p or R_p. The next lemma restates this observation more precisely. Let M_p be the point on the opposite side of $\overline{L_pR_p}$ from p which is at distance I from both L_p and R_p. Note that $P - \partial P$ is the interior of P.

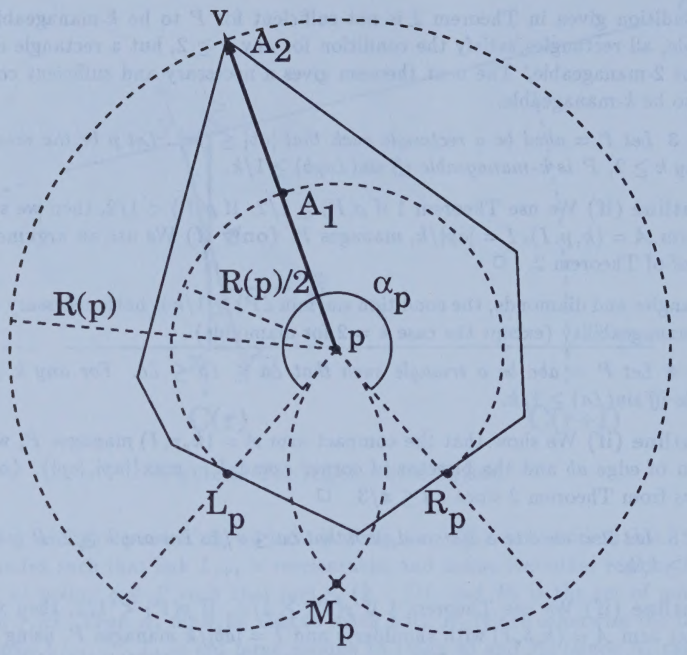


Figure 6: Illustration for Lemma 4.

Lemma 4 *Suppose that* $\rho(P) < 1/2$. *P is managed by the arm* $\mathcal{A} = (2, p, I)$, $I = R(p)/2$, *iff* $\alpha_p \geq \pi$ *and* $M_p \notin P - \partial P$.

Proof Outline Let C be the arc of A_1 having L_p and R_p as endpoints. Clearly, a point $p \in P$ is reachable by arm $\mathcal{A}$ (starting from a configuration in which $\mathcal{A}$ is fully stretched) iff $|pq| = I$ for some $q \in C$. It is obvious that the condition of this lemma is necessary and sufficient for all $p \in P$ to have such $q \in C$. □

By Theorem 1, Lemmas 3 and 4, we obtain:

Theorem 6 *Let P be a convex polygon. P is 2-manageable iff either*

1. $\rho(P) \geq 1/2$, *or*
2. $\rho(P) < 1/2$ *and for some* $p \in P$, $\alpha_p \geq \pi$ *and* $M_p \notin P - \partial P$.

The problem of computing the roundness $\rho(P)$ and fiding a point p which maximizes the ratio $r(p)/R(p)$ has been discussed in Section 4. In the following we consider the problem of finding p satisfying the second condition of Theorem 6.

Recall that $V(P)$ is the furthest-neighbor Voronoi diagram for the set of corners of P. The following lemma states that only the points in $V(P)$ need be considered when we look for a point satisfying the second condition of Theorem 6.

Lemma 5 *Suppose that* $\rho(P) < 1/2$ *and the arm* $\mathcal{A} = (2, p, R(p)/2)$ *manages P. Suppose that* $p \notin V(P)$, *and let v the corner of P furthest from p. Then the arm* $\mathcal{A}' = (2, q, R(q)/2)$ *manages P, where q is the point on* $\overline{pv}$ *closest to p such that* $q \in V(P)$.

Furthermore, we can prove the following lemma.

Lemma 6 *Suppose that $\rho(P) < 1/2$. Let $\overline{xy}$ be an edge of $V(P) \cap P$, where $u, v \in \gamma(r)$ for every $r \in \overline{xy}$ and $|xu| < |yu|$. Suppose that the arm $\mathcal{A} = (2, p, R(p)/2)$ manages P, where $p \in \overline{xy}$. Let q be the point in $\overline{xy}$ closest to x such that for each edge e of P between u and v (i.e., e is in the wedge determined by $\overline{pu}$ and $\overline{pv}$), the distance between q and e is at least $|qu|/2$. Then the arm $\mathcal{A}' = (2, q, R(q)/2)$ manages P.*

Using Lemmas 5 and 6, we can find a point p satisfying the second condition of Theorem 6 in $O(n^2)$ time, where n is the number of corners of P. We construct $V(P)$ in $O(n)$ time, and then for each edge e of $V(P) \cap P$, (1) find point q satisfying the condition of Lemma 6, (2) find point M_q for this q, and (3) test whether M_q lies in the interior of P, all in $O(n)$ time. We do this for each of the $O(n)$ edges of $V(P) \cap P$ and test whether any of the points q satisfies the second condition of Theorem 6. As we already discussed, the first condition of Theorem 6 can be tested in $O(n \log n + m)$ time, where m is the total number of intersections of $M(P)$ and $V(P)$. Note that $m = O(n^2)$. The following theorem summarizes the discussion given above.

Theorem 7 *For an n-sided convex polygon P, whether P is 2-manageable can be tested and if so, a 2-link arm which manages P can be found in $O(n^2)$ time.*

Now we return to the 2-manageability of diamonds.

Theorem 8 *Let $P = abcd$ be a diamond such that $\angle a \le \pi/2$. P is 2-manageable iff* $\tan(\angle a) \ge (1 + \sqrt{3})/2$.

Proof Outline If P is 2-manageable, then by Lemmas 3, 5 and symmetry, P is managed by a compact arm $\mathcal{A} = (2, p, I)$ for some $p \in \overline{bS}$ and $I = |ap|/2$, where $S = \overline{ac} \cap \overline{bd}$ is the center of P. We prove the claim by analyzing a critical situation in which point M_p coincides with b. ☐

6 Upper and Lower Bounds on the Number of Links

For a convex polygon P, let $\kappa(P)$ be the minimum number of links needed to manage P, i.e., $\kappa(P)$ is the smallest integer k such that P is k-manageable. The problem of designing an arm which manages P having exactly $\kappa(P)$ links is still open. In this section we prove a lower bound and an upper bound on $\kappa(P)$ using a parameter related to the roundness $\rho(P)$. Since the two bounds can differ by at most one, the proof of the upper bound gives us a method for constructing an arm for managing P having at most $\kappa(P) + 1$ links.

Let P be a convex polygon. As we did in Section 3, let $p_0 \in P$ be the unique point such that (1) $\rho(P) = r(p_0)/R(p_0)$ and (2) $R(p_0) \le R(q)$ for all q such that $\rho(P) = r(q)/R(q)$. Let C_0 be the circle with center p_0 having radius $r(p_0)$. Let θ be the apex angle of the wedge determined by two semi-infinite rays supporting C_0 which emanate from a corner in $\gamma(p_0)$. Note that θ is the acute angle such that $\sin\frac{\theta}{2} = r(p_0)/R(p_0) = \rho(P)$. In the following, we assume that $\theta < \pi/3$, since otherwise $\rho(P) \ge 1/2$ and P is 2-manageable by Theorem 1. The next theorem gives a lower bound and an upper bound on $\kappa(P)$.

Theorem 9 $\lceil 1/\tan\theta \rceil \le \kappa(P) \le \max\{3, \lceil \sqrt{5 - 4\cos\theta}/\sin\theta \rceil\}$.

The proof of the lower bound is similar to but more involved than that of Theorem 2. In the proof of the upper bound, we give a construction of a compact arm whose shoulder is in $\partial P \cap C_0$ and which manages P using $\max\{3, \lceil \sqrt{5 - 4\cos\theta}/\sin\theta \rceil\}$ links. The arm can be constructed in linear time, once p_0 is found. By analyzing the two bounds given in Theorem 9, we obtain:

Corollary 1 *The arm constructed in the proof of Theorem 9 has at most $\kappa(P) + 1$ links.*

Note that since $\sin\frac{\theta}{2} = \rho(P)$, $\sqrt{5 - 4\cos\theta}/\sin\theta = \frac{1}{2\rho(P)}\sqrt{1 + 9/(1/\rho(P)^2 - 1)}$, which approaches $1/(2\rho(P))$ as $\rho(P)$ becomes smaller. So an arm constructed in the proof of Theorem 1 can have nearly twice as many links as an optimal arm.

7 Concluding Remarks

We have considered the problem of designing a multi-link robot arm which manages a given convex polygon P. We have obtained, among other results, (1) a complete characterization of polygons which can be managed by a 2-link arm, and (2) almost matching upper and lower bounds on the number of links needed to manage a general convex polygon. Perhaps the most important open problem is to find a necessary and sufficient condition for a convex polygon to be k-manageable for a given $k \geq 3$. The related complexity issues must also be discussed. Another interesting problem is to improve the $O(n^2)$ time complexity proved in Section 5 for determining the 2-manageability of a given polygon. Also, note that all the arms we have constructed in the proofs are compact. We conjecture that for any k, all k-manageable polygons can be managed by a compact k-link arm. (This is true for $k = 2$, by Lemma 3.)

References

[1] A. Aggarwal, L. J. Guibas, J. Saxe and P. W. Shor, "A linear time algorithm for computing the Voronoi diagram of a convex polygon," *Proc. 19th Ann. ACM Symp. on Theory of Computing*, 1987, pp. 39–45.

[2] B. Chazelle and H. Edelsbrunner, "An optimal algorithm for intersecting line segments in the plane," *Proc. 29th Symp. on the Foundations of Computer Science*, 1988, pp. 590–600.

[3] H. Ebara, N. Fukuyama, H. Nakano and Y. Nakanishi, "A practical algorithm for computing the roundness," submitted.

[4] J. Hopcroft, D. Joseph and S. Whitesides, "On the movement of robot arms in 2-dimensional bounded regions," *SIAM J. Computing 14*, No. 2, 1985, pp. 315–333.

[5] V. Kantabutra, "Motions of a short-linked robot arm in a square," to appear in *Discrete and Computational Geometry.*

[6] V. Kantabutra, "Reaching a point with an unanchored robot arm in a square," manuscript, 1989.

[7] V. Kantabutra and S. R. Kosaraju, "New algorithms for multilink robot arms," *J. Computer and System Sciences 32*, 1986, pp. 136–153.

[8] V. B. Le and D. T. Lee, "Out-of-roundness problem revisited," *IEEE Trans. Pattern Analysis and Machine Intelligence 13*, No. 3, 1991, pp. 217–223.

[9] D. T. Lee, "Medial axis transformation of a planar shape," *IEEE Trans. Pattern Analysis and Machine Intelligence PAMI-4*, No. 4, 1982, pp. 363–369.

[10] F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*, Springer-Verlag, 1985.

[11] J. H. Rief, "Complexity of the mover's problem and generalizations (extended abstract)," *Proc. 20th Symp. on the Foundations of Computer Science*, 1979, pp. 421–427.

[12] C.-K. Yap, "Algorithmic motion planning," in *Advances in Robotics Vol. 1: Algorithmic and Geometric Issues*, Schwartz and Yap, eds., Lawrence Erlbaum, 1987.

Management of Semiconductor Manufacturing Using Discrete Event Simulation

William Grey
Sarah Hood
IBM Thomas J. Watson Research Center
Yorktown Heights, NY 10598
(914) 945-3660

Abstract

A discrete event simulation model was developed to represent semiconductor and thin films manufacturing lines. A configuration file is used as an input to the model to describe the line, products being manufactured, and modeling assumptions. The model was validated in a VLSI wafer line, where it accurately simulated cycle times and WIP. It was used to assess WIP management strategies, determine optimal line loading, and as a complement to the tool planning process.

Introduction

The Semiconductor and Thin Films Line Model is a highly detailed discrete event simulation model for managing the semiconductor manufacturing process. It provides a modular and reconfigurable representation of manufacturing lines to help study scheduling and control policies to reduce cycle time and increase efficiency. It may be used for existing lines to evaluate proposed changes, or to develop tool and staffing plans when designing new lines.

We will show why a detailed model is particularly well suited to represent the complex task of managing the semiconductor manufacturing process. Results will be presented demonstrating the use of this model as a decision support tool in a number of existing semiconductor lines, and as a planning tool for a new thin films packaging line.

Semiconductor Line Characteristics

The long cycle times and complex dynamics of semiconductor manufacturing make it a particularly difficult process to manage effectively. Process flows are frequently comprised of hundreds of steps, which are performed on dozens of different tools. Routings are highly reentrant: a job may visit a tool group ten times or more before processing is completed (Figure 1). Furthermore, multiple products are frequently manufactured in the same facility.

The competitive market drives the frequent introduction of new products with smaller, tighter tolerances. This provides an impetus to reduce cycle time, in order to accelerate the learning process, respond more rapidly to changing demand in the marketplace, and reduce exposure to contaminating particles during the device fabrication process. At the same time, the drive to smaller feature sizes makes the goal of cycle time reduction increasingly difficult. Process flows are becoming longer, and more complex. Tools may be operated at the limit of their specifications, which may introduce significant variability, reduce yield, and increase the need for rework.

The high cost of equipment, and the specialized environment required for semiconductor manufacturing make it essential to effectively balance a line, and to improve line loading, lot size, scheduling policies, and tool and operator availability. However, the long cycle times and the complex dynamics of semiconductor manufacturing make it difficult (and expensive) to experiment on the line itself. Analytic models such as linear programming and mean value analysis, simulations with many simplifying assumptions, and manufacturing experience have been employed to reduce cycle time. A detailed, discrete event simulation model is an alternative which can supply further insight by explicitly representing rework, sendahead (test) wafers, and operators. Since it provides a "virtual line" on which to experiment, it can serve as a valuable tool in assessing the impact of competing line management strategies.

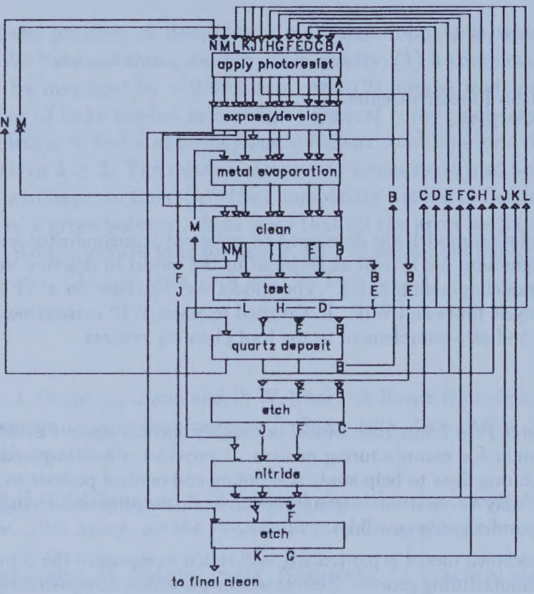


Figure 1. Job Flow in a Semiconductor Manufacturing Line

Model Description

The goal in designing this model was to represent semiconductor and thin films manufacturing lines in considerable detail, yet provide a flexible interface for specifying different line and product configurations. The discrete event simulation language SIMAN (4) was used, but the requirements for a flexible user interface led to a model structured differently than the conventional SIMAN model. In SIMAN, the characteristics and relationships of the system's components are described in the model frame, and specific variable values are defined in the experiment frame. For a complex model, however, considerable expertise is required to make even simple changes in the model's structure or its inputs. Furthermore, recompilation and relinking are necessary whenever the model or experiment frame are updated, incurring considerable overhead.

Improved flexibility and performance were obtained by employing a customized configuration file as the user interface to the model. This ASCII file, which is read in at the beginning of the simulation run, contains all the input data describing the line, and the information necessary to conduct simulation runs. These include input control parameters, the input schedule, shift and simulation run (time) parameters, expediting policies and parameters, rework policies and parameters, dispatching (job selection) policies and parameters, product types, process flows and rework sequences (routings), tool group parameters (capacity changes, failures, setups, operator ratio), and operator pool parameters (capacity changes, staffing levels).

To implement this architecture in SIMAN, the model frame uses submodels called station macros to represent tool groups and tools. A station macro is similar to a subroutine: it is a generic set of logical statements which act on the parameters passed to it. Each macro consists of a set of individual stations, which represent related resources such as tool groups, or tools within a tool group. These macros call events to execute FORTRAN subroutines which make determinations such as how to assign operators, which job to process next, and how to route a job needing rework. The FORTRAN subroutines depict detailed aspects of the system and are easily manipulated by specifying options in the configuration file.

In the model, two macros represent the flow of jobs through the manufacturing process: one which contains the code which applies to tool groups, and another which comprises the code for individual tools within a tool group. Station macros were also used for three tool-based external events: preventive maintenance, major setups, and failures. With tools numbering in the thousands, this design greatly reduces the overhead of these events by only activating the control entities for the duration of the events. External events are independent occurrences which affect the system. Jobs do not flow through these parts of the model; instead, these events affect the system's performance by changing resource availability or job attributes. These events were implemented as separate sections of model code with their own control entities. The time between execution of these events and their durations are specified in the configuration file, and may be random. In addition to the three tool-based external events which use macros, there are four external events which did not require the macro structure: initialization of variables, assignment of a roving operator group called the flying squad, expediting jobs, and preloading the queues at the beginning of a simulation run.

Certain aspects of the simulation were represented with FORTRAN subroutines invoked within the SIMAN model; these include the input schedule, shift (time) events, operator assignments, resource (tool and operator) capacity changes, expediting policies, rework policies, and dispatching rules. These FORTRAN subroutines are easily manipulated by turning options on or off in the configuration file. Our model also uses FORTRAN user functions to replace many entity attributes, resulting in a flexible structure with reduced memory requirements.

The operator module, for example, is designed so that operators may be assigned to more than one tool, and may perform up to three different tasks at a single tool: load, monitor, and unload. By assigning individual operators to specific tools, the simulation model reflects the fact that operators may monitor up to N tools, but perform load and unload on only one tool at a time. A secondary operator pool called the flying squad is also reassigned at the start of every shift to tool groups to help out the other operator pools. Long term capacity changes to tool groups, operator pools, and the flying squad are applied on a monthly basis.

The code for expediting jobs describes three policies for altering a job's priority: random expediting, end of month expediting, and expediting based on demand. Random expediting reassigns a job's priority at random intervals. The job's priority may be increased or decreased, and the affected job may reside at any stage in its sequence. End of month expediting represents the push to ship product at the end of the accounting month. A few days before the end of the month, those jobs which can be completed by the end of the month are assigned priorities above all other jobs. Expediting based on demand is an event which occurs at regular intervals. This event considers all jobs and compares their current progress with their due dates. Based on these comparisons, the jobs' priorities are either increased or decreased. Each of these three policies may be turned on or off, independently of each other, through the configuration file.

The rework code represents various possible policies on how to split bad wafers out of a job and how to recombine reworked wafers with original good wafers.

The dispatching code embodies the various rules used to determine which job to process next when a tool and one of its operators becomes available. Waiting jobs are considered for processing in the following order: jobs with a special "super" priority are selected first, followed by "split" jobs, in which the first part of the job has already been processed, and the remainder has been forced to wait because of tool batch size limitations. If no such special jobs require processing, lots are selected depending on an option specified in the the configuration file: either on a strict priority basis, or by selecting the job that will minimize tool setup. Options are also available to reject a selected job immediately if its downstream tool group's queue is full, or if it cannot be completely processed before the end of the current shift. Once a job has been selected, it is checked for special processing requirements or options. These include use of a sendahead (test) wafer, in which part of the job is processed and inspected prior to processing the remainder; batching with other jobs; or splitting jobs whose lot size exceeds the batching capability of a tool.

The FORTRAN code is divided into approximately sixty subroutines; this modularity facilitates enhancing and maintaining the model. For example, all code relating to job expediting is grouped into three subroutines, thus limiting the number of modules that must be referenced when making modifications.

Validation

The Semiconductor and Thin Films Line Model was validated using data from an IBM VLSI wafer line in East Fishkill, New York. Validation was accomplished through a process of updating and refining model data until model results such as cycle time, Work-In-Process (WIP), and tool utilization compared favorably with actual line measurements.

Successful model validation, of course, largely depends on the ability to obtain accurate input data. The Semiconductor and Thin Films Line Model requires four main bodies of input data: product releases, process sequences, tool group parameters, and operator parameters. Because of the inherent complexity of wafer fabrication, assembling these four main categories of data alone required the collection of thousands of input parameters.

To expedite the initial construction of the input data file and obtain preliminary results, a combination of planned and actual input data was used, since actual data was not always readily available in a format suitable for direct input into the model database. In this case, planning parameters were used to complete the tool group input requirements such as set-up and maintenance downtimes, and the number of tools online. Further, it was assumed that each available tool was fully manned at the planned operator-to-machine ratio at all times, except lunches and breaks. This was done to identify strictly tool-related capacity bottlenecks first, without regard to possible manpower constraints.

By contrast, primarily actual data was used to complete the product release and process sequence databases. Actual product release volumes were obtained from the manufacturing control center, although no actual priorities were used at this point. Actual process flows, processing times, and batching rules were obtained from routings and manufacturing operators.

This preliminary input configuration produced results which were generally quite optimistic in relation to actual line performance. Model cycle times and WIP, for instance, were initially lower than actuals by a factor of two.

As time progressed, more actual data such as downtimes for gating tools, manpower deployment, rework percentages, and scheduling priorities were obtained and incorporated into the model, replacing plan data. With each subsequent model configuration update, the simulation results more closely approached actual line measurements, until eventually our model was satisfactorily validated.

The importance of a detailed representation of the dynamics of semiconductor manufacturing is shown in Figure 2, which illustrates the effect that simplifying assumptions can have on model accuracy. In the detailed case, wafer rework was modeled explicitly, and actual data was used to represent operator break profiles and tool staffing levels. In the simplified case, rework was modeled by increasing raw process times, and operators were not treated as a constraint. As the figure shows, the detailed model was much better at modeling sector cycle time, and total WIP in the line. Actual data was not available for average WIP at tool groups, but the detailed model captured the variability typical of tool queues, with spikes in the data at bottleneck tool groups 1, 4 and 7.

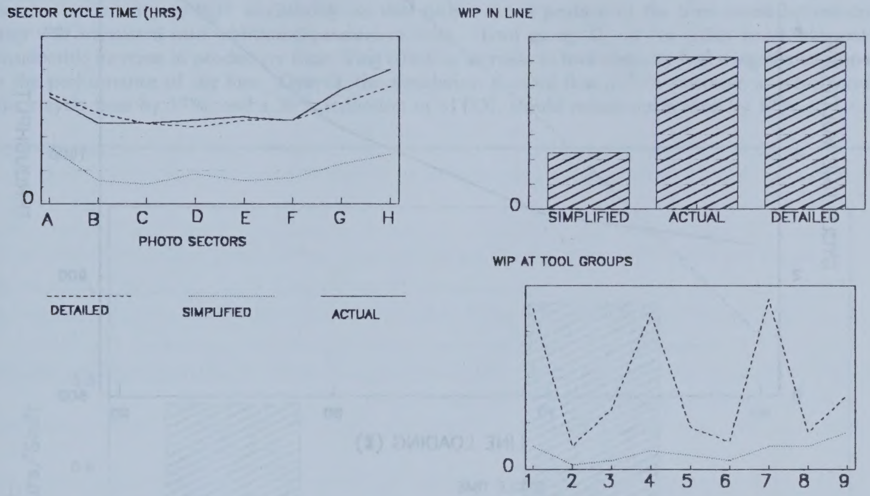


Figure 2. Comparison of Detailed and Simplified Models

WIP Management and Line Loading

One of the areas in which a line model can have its greatest impact is in assessing WIP management strategies, and tuning line loading. Release policies, which determine how lots are released into the line, and dispatching rules, which are used to decide which lot to process next when multiple lots are competing for the same tool or operator, can have a major influence on cycle time and throughput (1,2,5). Direct experimentation to compare competing strategies is not feasible, since the long cycle times characteristic of semiconductor lines make it too costly and time-consuming to perturb an actual line, and then study the effect on line performance.

In the model, however, parameters specifying job release rates, release policies, dispatching rules, and expediting policies can be easily changed. Years of production time can then be simulated in hours, without interfering with the operations of the actual line. Figure 3 shows the relationship between two performance measures, cycle time and throughput. Line loading is represented as a percentage of a theoretical maximum number of wafer starts per day. The cycle time multiplier is the ratio of actual cycle time to theoretical raw process time. According to these results, this manufacturing line should not be loaded at more than 80 percent of its maximum capacity, since there is no gain in throughput, and cycle time increases dramatically. If particularly short turnaround times are required, one may even wish to reduce the starts further and sacrifice throughput for faster cycle time turns, and therefore more learning cycles. Miller (3) has calculated this characteristic curve for a semiconductor development line.

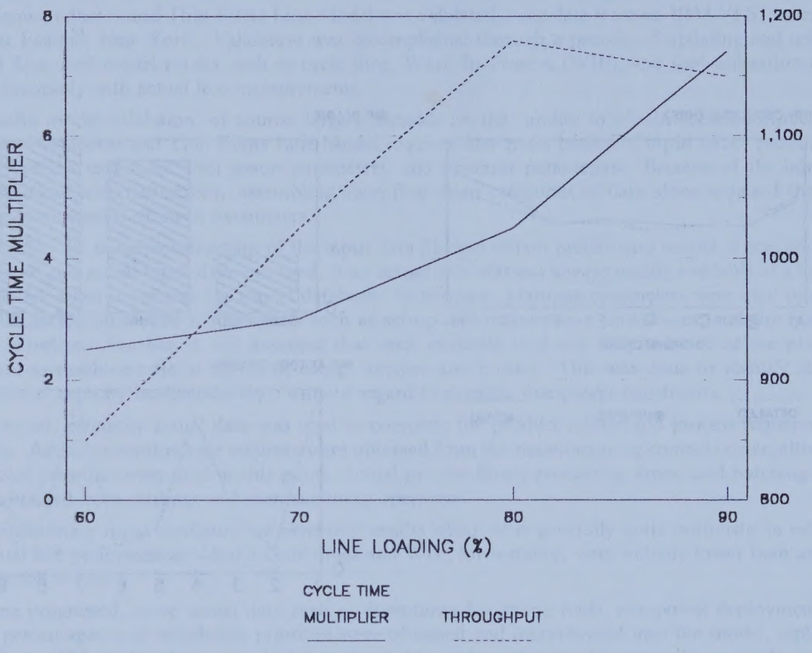


Figure 3. Cycle Time and Throughput vs. Line Loading

Tool Planning and Line Balancing

This model provides a valuable complement to conventional tool planning. It represents the effects of operators, the randomness of events such as tool failures, and the dynamics of job flow, while the tool plan deals only with mean values for tool parameters. However, the model does not determine the optimal number of tools. Therefore a tool plan is still necessary to provide an initial estimate for the number of tools to input to the model.

Our model was used in the tool planning process for a Packaging Line. The objectives of the initial characterization were to identify bottleneck tool groups, and determine line throughput capability and WIP, while still maintaining acceptable cycle times. Two bottleneck operations, were identified. Use of these operations in process routines were reviewed, to identify unnecessary steps, and steps with excessive process times. After making these changes, a release rate was determined which provided desired WIP within acceptable cycle time limits. (It was felt that a minimum level of WIP was required to provide adequate process learning). This release rate was used for subsequent analyses.

Three different process alternatives were analyzed; two basic results were obtained. The first suggested that the bake and cure process held the greatest potential for cycle time reduction. The second suggested that regardless of which processes were used, operator constraints, i.e. breaks, lunch, and meetings, had a significant impact on cycle time, in some instances doubling it.

An additional analysis evaluated the capacity required for the metal sputter tool. The original tool plan specified a sputter tool with a capacity of nine substrates. The simulation showed that there was essentially no change in cycle time until the tool batch size was reduced to one; even then, the increase in cycle time was only 13%. Based on the results of the simulation, a four substrate tool is being purchased at a substantial savings.

The usual response to the need for more tool capacity is to add tools, but in some cases it is also possible to reduce setup, maintenance or down time to make more time available for processing jobs. Our model was used to simulate the effect of reducing setup time and mean time off-line (MTOL, time waiting for repair, plus repair time) by 50% for tools in an IBM manufacturing line. The study showed that the impact of this change was not uniform across tool groups. Figure 4 compares simulation results for two bottleneck tools for which setup time was reduced by 50%. Tool group A was gated by operator availability or WIP availability, so that only a small portion of the time saved by reducing setup was translated into additional productive time. Tool group B, on the other hand, showed a considerable increase in productive time. This effective increase in tool capacity had a significant impact on the performance of the line. Overall, the simulation showed that a 50% decrease in setup would reduce cycle time by 37%, and a 50% reduction in MTOL would reduce cycle time by 17%.

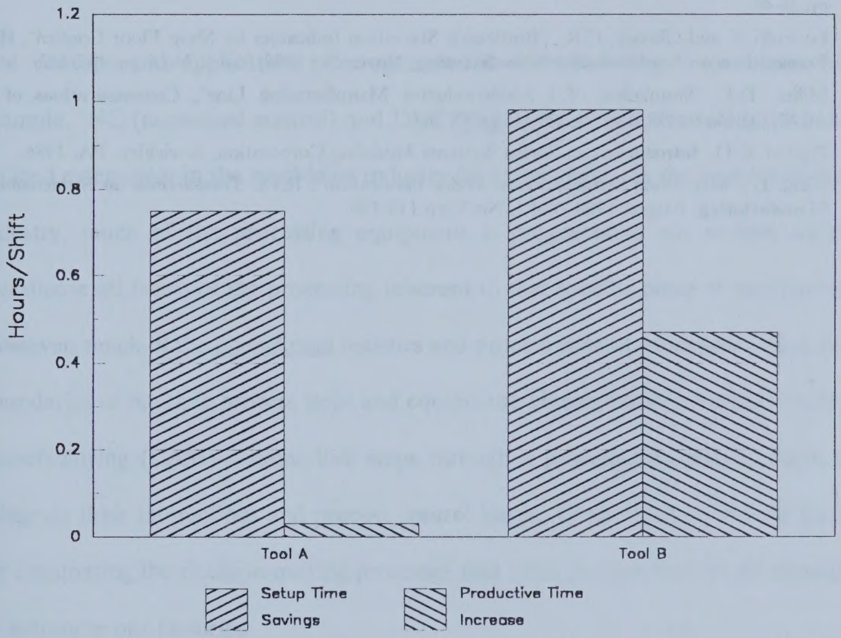


Figure 4. Effect of Tool Availability on Productive Time

Summary

A discrete event simulation model was developed to represent semiconductor and thin films manufacturing lines in considerable detail. It was written in the SIMAN modeling language, and uses a large number of FORTRAN modules to simulate the dynamics of semiconductor production. A configuration file is used as an input to the model: it describes the line and products being manufactured, and supplies details about the simulation run and its assumptions. The model was validated in a VLSI wafer line, where it accurately represented cycle times and WIP.

The model was used to determine optimal loading in a manufacturing line, and to analyze line WIP, tool WIP, and sector cycle time. It was found that explicitly representing details such as rework, test wafers, and operators significantly improved the accuracy of the model.

The model was also used in the tool planning process for a Packaging Line to identify bottlenecks, and assess line performance and capacity. Bottleneck processes and tools requiring additional staffing were identified, and tooling changes were recommended.

Acknowledgements

The authors would like to acknowledge and thank Robert R. Leavitt for his contributions to the Packaging line study, Calvin L. Clemons for his contribution to the validation process and line loading study, and Dave Robideau for his role on the specifications team.

References

1. Glassey, C.R. and Resende, M.G.C., "Closed-Loop Job Release Control for VLSI Circuit Manufacturing", **IEEE Transactions on Semiconductor Manufacturing**, February 1988, Vol.1, No.1, pp.36-46.

2. Lozinski, C and Glassey, C.R., "Bottleneck Starvation Indicators for Shop Floor Control", **IEEE Transactions on Semiconductor Manufacturing**, November 1988, Vol.1, No.4, pp.147-153.

3. Miller, D.J., "Simulation of a Semiconductor Manufacturing Line", **Communications of the ACM**, October 1990, Vol.33, No.10, pp.98-108.

4. Pegden, C.D., **Introduction to Siman**, Systems Modeling Corporation, Sewickley, PA, 1986.

5. Wein, L., "Scheduling Semiconductor Wafer Fabrication", **IEEE Transactions on Semiconductor Manufacturing**, August 1988, Vol.1, No.3, pp.115-130.

The Programmable Factory

Randal E. Null, Ph.D.
Intel Corporation
5000 West Chandler Boulevard
Chandler, AZ 85226

The concept and application of programmable manufacturing is not new; for example, NC (numerical control) and DNC (digital numerical control) have been utilized extensively in the machining industry for many years. In the semiconductor industry, much of the processing equipment is programmed via recipes which coordinate all facets of the processing inherent to that specific piece of equipment. However, much of the operational logistics and process control procedures cross the boundaries of multiple process steps and equipment. Hence, today's computer aided manufacturing (CAM) systems link steps through a process flow but do little to integrate their interactions and process control loops. In short, there is little basis for automating the decision-making processes that cross process lines as we attempt to automate our factories.

Traditional shop floor control systems in a CAM-operated factory are focused on work-in-process tracking. Ancillary functions such as specifications, equipment, and scheduling play a secondary role in their support of the material movement activity. However, as we look forward towards a next- generation Computer Integrated Manufacturing (CIM) system in peopleless factories, the critical questions revolve around the rules by which a factory operates rather than as the flow of material. A Manufacturing Rules Management (MRM) system must become the integrating agent across a distributed network of server functions. A suite of services to define, manage, execute, and monitor the rules truly represents the glue by which the "islands of automation" are integrated. In essence, one should have available a vehicle to program the characteristics and decision-making processes by which a factory is to operate.

Such a programmatic interface for factories solves a multitude of problems faced by factory management today. In particular, semiconductor factories operate in an extremely volatile environment and must react to dynamic changes in a rapid manner. A single interface to alter the factory profile and test the validity of those alterations will ensure rapid, accurate modification. Additionally, given the cost of new factories in this industry, the need to ramp to full production in the shortest cycle possible is critical in achieving an acceptable return on assets. Concurrent work in the area of knowledge capture and representation in the R&D facilities provides a mechanism to program the production factory at start-up and smooth the technology transfer.

This talk will investigate one possible approach to the "programmable factory" and discuss significant hurdles that must be overcome to implement such a system. At Intel, we recognize that the technology is secondary to the enormous paradigm shift in operational mentality that must be achieved. Indeed, systems can only replicate faulty logic; they cannot correct it.

EQUI-VARIABILITY GRAPH APPROACH
FOR
MODELING OF MANUFACTURING SYSTEMS

By SHU LI

Intel Corporation
5000 W. Chandler Blvd.
Chandler, AZ 85226

September, 1991

Abstract: An everyday problem facing a modern manufacturing factory is to identify bottlenecks, dealing with production variabilities for predictable delivery, and forecast the real capability of the manufacturing facility. A very practical methodology is introduced, which has been used in real life semiconductor factories. The method also provides a new dimension and philosophy for mathematical modeling of large complex systems.

What Is the Problem?

Picture yourself in a manufacturing environment, say a semiconductor fabrication process, you face with hundreds of steps of operations, very complex networks of machines shared by operations in the front and the end of the processes, and many different kinds of processes and products. You would like to produce as predictable as possible so that the orders accepted are to be shipped on time, and the maximum inventory turns are achieved for the required output. You want to accept as many orders as possible so as not to miss the opportunity loss of market shares to your competitors. You would like to devote your valuable limited resources to tackle your weakest spots, i.e. the bottlenecks. On the other hand, although you pray for good performance of the machines and less variabilities in the production, almost every minutes, something unexpected may happen: machine problems shut down the line for hours or even days; process problems due to marginal technology (e.g. the gate oxide for semiconductor devices may require a perfect layer of oxide of thickness of 150 Angstroms -- one three thousandth of the diameter of a human hair); human errors (e.g. an operator could accidentally drop chemical bottles which leads to evacuation of the factory); variabilities among different shifts on staffing, prioritization strategy, break coverage, ergonomic considerations at the operation; incoming vendor material defects; facility variabilities (e.g. power glitch could dump high vacuum machines, water contamination could destroy the quality of the processes); etc. Given all complexity, it is usually very difficult to model a large system using existing methods. The modeling effort either evolve into a huge data collection/validation and endless details to trade-off, or a large amount of unjustifiable and unrealistic assumptions. Thus it is hardly seen any large scale system modeling in the real world. Most of the factories are still using static models such as MRP or CEFAS or even simply "gut feel" for capacity and loading purpose. However, these static models usually lead to the wrong conclusions.

To show the danger, a real life example of the dilemma of the static model versus the actual production capacity is discussed here. Three similar machines are responsible to run 10 different kinds of operations. Based on the average demand rate of these operations, and the average processing rate of the machine (the ratio of the two is the so-called loading intensity), it was concluded that only 1.8 machines are needed. The third one could be removed, and save space for other purposes. The execution plan called for a trial period of five weeks with one machine hot bagged. At the end of the time period, the following data was reported:

Work week	1	2	3	4	5	Average
Weekly goal	1466	2673	2673	2673	2673	2431.6
Actual output	2011	2871	2851	2832	2919	2696.8
Delta	545	198	178	159	246	265.2

It is thus concluded that two machines could actually handle the demand very well. On the other hand, manufacturing people who run the machines everyday, argue for insufficient capacity. Their observation is that the inventory piles up in front of these machines. A little bit deeper investigation shows the following data:

Work week	1	2	3	4	5	Average
Weekly ave cycle time (hrs.)	8.5	6.0	21.0	26.2	16.5	15.64
Weekly goal (4xtheoretical)	6.0	6.0	6.0	6.0	6.0	6.0
Delta	-2.5	0.0	-15.0	-20.2	-10.5	-9.64

It is thus impossible to satisfy the due date commitment (based on weekly cycle time goals). Factory inventory turns would be hurt, total inventory would grow, cycle time would increase, and customer shipment would be delayed.

What this story tells us is that static models of capacity using various first order average statistics is not dependable to study factory capacity. Many similar examples exist. On the other hand, there are known difficulties using models such as queueing, simulation, and petri-net. Hence, what is the alternative?

A Simple but Useful Alternatives to Existing Modeling Methods

1. Results from queuing theory:

For a manufacturing entity, no matter it is a machine, a group of machines, or a factory, we could consider it as a service center. Items arrive at the center, queued up there, waiting for processing. The center serves the job according to some predetermined rules or orders. The finished items leave the center for further activities. Assuming that items arrive in a average rate of λ, and that each machine has a average processing rate of μ. From the results of G/G/1 queue ([1],[2]), assuming $\lambda<\mu$, the average number of items in the system N is approximately given by Kingman's formula:

$$N = \rho \left[1 + \frac{\rho V}{2(1-\rho)} \right], \text{ where } V = \lambda^2 \sigma_I^2 + \mu^2 \sigma_P^2$$

where $\rho=\lambda/\mu$, and σ_I and σ_P are the standard deviations of the interarrival and processing times, respectively. This is the universal relation among ρ, the loading intensity, N, the inventory, and V, the variability. Based on this model, we could draw a graph. We call it the Equi-Variability Graph as shown below:

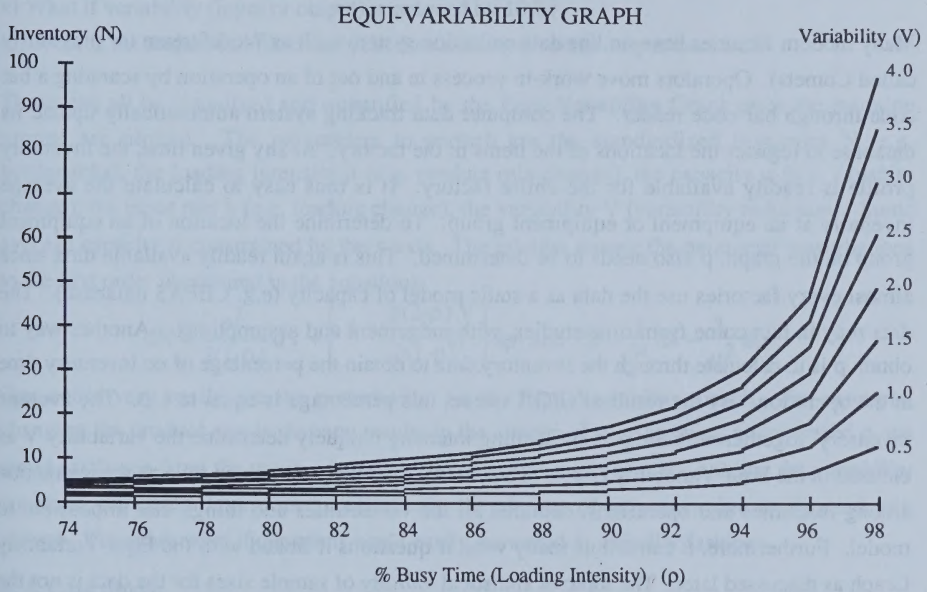


Fig. 1

2. Explanation of the above Dilemma.

Now the dilemma can be very easily explained. The static study based on loading intensity only corresponds to the x-axis of the Equi-Variability Graph. However, inventory, which is the real indicator of a bottleneck situation, is determined by both the x-axis and the y-axis, i.e. the Variability of the system. As another real life example, a group of 45 identical machines shows a loading intensity ρ of 98%. But due to low variability, inventory is at the level of less than 10. On the other hand, the machine group shown earlier has a loading intensity of 90% with two machines, but inventory of nearly 20 due to large variabilities. The later is thus much more of a concern than the former.

3. Let the Data Serve As Network Connectors.

The solution of network of queues of G/G/1 type requires many simplified assumptions and approximations ([3]-[9]) to be computationaly feasible. Furthermore, modeling all the uncertainties, which are composed of all of the variabilities, is an almost impossible task in real life manufacturing. To make the model capable of representing huge network of interactions of machines, people, materials, and production management, real data is used as network connectors, which not only eliminates conventional modeling effort and drawbacks, but also builds realistically credible results. To utilize the graph, we need two types of data, i.e. the average inventory and the loading intensity ρ. It is impossible to directly determine V in a real life situation.

Many modern factories have on line data collection system such as WorkStream (or previously called Comets). Operators move work-in-process in and out of an operation by scanning a bar code through bar code reader. The computer data tracking system automatically update its database to register the locations of the items in the factory. At any given time, the inventory profile is readily available for the entire factory. It is thus easy to calculate the average inventory at an equipment or equipment group. To determine the location of an equipment group on the graph, ρ also needs to be determined. This is again readily available data since almost every factories use the data as a static model of capacity (e.g. CEFAS database). The data may in fact come from time studies, with judgement and assumptions. Another way to obtain ρ is to calculate through the inventory data to obtain the percentage of no inventory time at the operation. By the result of G/G/1 queue, this percentage is equal to $1-\rho$. The average inventory, together with the percent loading intensity uniquely determine the variability V as dictated in the Equi-Variability Graph. Thus the average inventory alone reflects the interaction among machines and operations, captures all the variabilities and things that impossible to model. Furthermore, it can tell us many what if questions if linked with the Equi-Variability Graph as discussed later. The topic of statistical validity of sample sizes for the data is not the main focus of this paper. There is ample amount of literature to cover the topic.

4. Standardization of Inventory.

Before we dive into a case study, there is one more thing to clarify. For G/G/1 to be applicable, the inventory needs to be standardized towards a single machine equivalent, and the moving batch size (i.e. batch size that items stay together to physically move out of the operation). Therefore, in plotting the Equi-Variability Graph, the inventory used is the standardized inventory defined as the follows:

$$\text{Standardized Inventory} = \frac{\text{average inventory}}{\text{average moving batch size x number of machines}}$$

5. What if Analysis.

The next step to use this model is to answer what if questions. Among the questions, the most important ones asked frequently by manufacturing managers and executives are:

i) Which is the biggest bottleneck operation -- the place to have highest return on investment? This is an important question to direct the process of on-going improvement.
ii) What is the effect on factory cycle time and inventory if demand is increased or decreased? This is a major question for production planning and control, as well as customer commitment.
iii) What if the product mix is changed? Setups and capacity may be impacted in this case.
iv) What if capacity is increased by 10%? Does demand be met or bottlenecks disappear?
v) What if variability (input or output) is reduced by 10%?
vi) What is the major road block to cycle time reduction versus output increase?

These can all be visualized and quantified by the Equi-Variability Graph once the machine groups are plotted. The parameters to perturb are the standardized inventory N (e.g. bottlenecks), the loading intensity ρ (e.g. product mix change), the capacity μ (e.g. capacity change), the input rate λ (e.g. loading change), the variability V (variability reduction). Static average capacity is constrained by the x-axis. The relation among the parameter perturbations to the first order is captured in the equations:

$$dN = \frac{\rho^2}{2(1-\rho)} dV + \left(1 + \frac{\rho(2-\rho) V}{2(1-\rho)^2} \right) d\rho \quad \text{and} \quad d\rho = \frac{1}{\mu} d\lambda - \frac{\lambda}{\mu^2} d\mu$$

One could very easily see the perturbation on the Equi-Variability Graph. For example, changing the product mix in demand results in the change of ρ. Assuming an increased ρ, we could easily see from the graph a increased inventory accordingly. By holding the variability constant, we could determine quantitatively the change of inventory due to demand mix change. The other what if questions could easily answered in a similar fashion.

6. A Case Study.

The factory in this case study is a large the state-of-the-art semiconductor fabrication facility which is capable to output approximately 1000 finished wafers (one wafers contains 200-1000 integrated circuits) per day. There are about 45 work centers consisting of about 200 machines, excluding inspection stations, auxiliary cleaning stations, etc.. More than 10 different process flows, each consisting of some 15 different semiconductor devices run through the line. Most of the operations require different setups for different devices or processes. The Equi-Variability Graph method above has been used to study the entire factory. All of the major equipment groups were considered in the model. For brevity, here we show the results of seven of the work centers. In the following table, the inventory and loading intensity are calculated from inventory data in a period of two and a half months, while the variability data is calculated according to the Kingman's formula.

Equipment Group	I	II	III	IV	V	VI
Standardized Inv.	1.57	15.37	1.5	2.1	2.34	5.15
Loading Intensity	78%	93%	91%	98%	85%	91%
Variability	0.6	2.2	1.4	0.05	0.6	0.7

This is also plotted on the Equi-Variability Graph (Fig.2)

Equi-Variability Graph (Case Study)

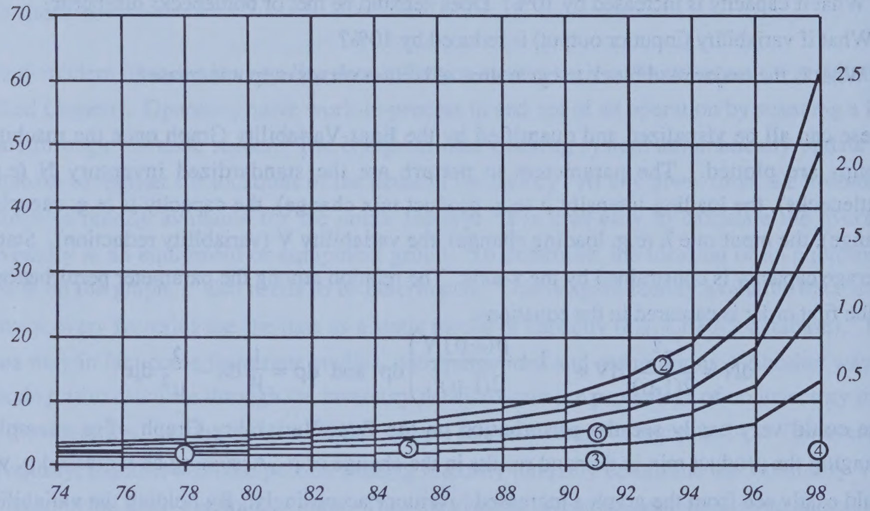


Fig.2

Among the many dimensions of conclusions from the model, the followings have been useful for the particular application:

i) Machine group 4, although previously were considered as the number one bottleneck for the present loading, it should not be of a big concern. The conclusion has been confirmed by the observation of the line and the gut feel of the production people. The reason that such a high loading intensity does not induce a bottleneck is apparently due to the low variability. However, if the potential market demand would increase the loading, this group could evolve into the number one bottleneck.

ii) Machine group 2 has the highest variability among the ones studied. It is also the one which accumulates the highest standard inventory in order to fulfil the required output. As a result, the cycle time at this group of machines must be high as dictated by Little's Law: $N = \lambda T$, where N is the average inventory, λ is the arrival rate, and T is the cycle time. Thus if the current loading and mix is stable, this would be the focal point for improvement. In fact, this group of machines are high vacuum equipment, with very low reliability, marginal process capability, long setup time, as well as high percentage of preventative maintenance time. The effort followed has dramatically reduced the variability, leading to a much smoother line and lower cycle time.

iii) To show at least one what if analysis, suppose that a future loading is predicted by the marketing and planing division that requires a 3% more output at group 2. For fixed capacity (i.e. μ) and fixed variability (i.e. V), inventory is almost going to double (see Fig.2). In fact, our experience shows even higher variability as the machines are more utilized due to increased down time. Therefore, the increase of the inventory could be even higher that 100%. Thus this group of machines becomes very sensitive to external demand. To know this is also key to production planning. If the loading is increased, and the planning cycle time is still the still the same as before, and no solid measurable improvement of the operations is underway, the customer orders would be in jeopardy due to longer cycle time.

The Concept of Bottleneck and the Process of On-going Improvement

The only true measure of bottlenecks is the inventory, which is the combined effects of both loading intensity and variabilities. Loading intensity represents the static part of the capacity, while variabilities represent the dynamic nature of the system. Equi-Variability Graph method can help identify these weakest operations easily. Most of the time, the bottlenecks are due to high variabilities because of setups, long preventative maintenance, unscheduled down time, operator errors, unbalanced line, large batch production, etc. Once these bottleneck operations are identified, more detailed study on the operation is required to determine the real cause of the variability. As the concept of improvement teams such as TQC is maturing, this new tool provides guidance for the troops to tackle the most leveraged spots. When the problems are resolved, the tool goes through another iteration of bottleneck study, and points out new directions for leveraged improvement. It is through this endless process of iterations, and relentless improvement efforts to focus on the most leveraged spots of the manufacturing systems that cycle time is brought to the minimum, inventory is greatly reduced, and productivity is dramatically increased.

References

[1] Kleinrock, L. (1975) "Queueing Systems", John Wiley & Sons.

[2] Gelenbe, E., and Pujolle, G. (1987) "Introduction to Queueing Networks", John Wiley & Sons.

[3] Gelenbe, E. and Pujolle, G. (1976) "The Behavior of a Single Queue in a General Queueing Network", *Acto Informatica*, 7, 123-136.

[4] Gelenbe, E. and Pujolle, G. (1977) "A Diffusion Model for Multiple Class Queueing Networks", Proc. of the 3rd International Symposium on Modeling and Performance Evaluation of Computer Systems, Bonn.

[5] Reiser, M. (1979) "Mean Value Analysis of Queueing Networks, A New Look at an Old Problem", *IBM Research Report* RC, 7228.

[6] Courtois, P.J. (1971) "On the Near Complete Decomposability of Network of Queues and of Stochastic Model of Multiprogramming Systems", Carnegie-Mellon University Research Report, Computer Science Dept.

[7] Courtois, P.J. (1975) "Decomposability, Instabilities and Saturation in Multiprogramming Systems" *CACM*, 18, 371-377.

[8] Labetoulle, J. and Pujolle, B. (1980) "Network of Queues", *IEEE Transaction on Software Engineering*, 6, 373-381.

[9] Kamath, M, Suri, R., and Sanders, J. (1988) "Analytical Performance Models for Closed Loop Flexible Assembly Systems", *International Journal of Flexible Manufacturing Systems.*

STATISTICAL CIRCUIT DESIGN CENTERING USING RESPONSE SURFACE METHODOLOGY

by
Mark R. Rencher
Bipolar Analog Division, Motorola
2100 E. Elliot Rd. Tempe, Az 85283

Abstract- Design for Manufacturability is the technique of predicting circuit performance (yield) by modeling the process variation. By predicting the yield, process and design engineers are able to improve a products quality by the technique called design centering.

Design centering has been a field of study since the mid 1970's. From that time there has been many approaches that provided optimization techniques to improve yield. However, these approaches were week in defining the region of acceptability.

To overcome these weaknesses a new method using MSTAT (Motorola STatistical Analysis Tool) to model process variation/yield and the use of RSM for design centering (yield improvement). The key to this new method is the ability to transform the specification into component parameter (design) space by defining the region of acceptability as a function of the response surface created by RSM.

I INTRODUCTION

Design for manufacturability is the ability to predict the changes in an IC's behavior with variations in the process and make adjustments to the design and/or process to improve the quality.

Historically design for manufacturability consisted of multi-pass silicon, worst case/best case simulations and/or Monte Carlo simulations. In each of these cases the costs involved were significant and the level of information to achieve a high quality (six sigma) were not attainable.

To achieve the high design quality that Motorola customers demand and to meet the mandate of six sigma quality, a design methodology embracing statistical circuit design, using MSTAT for yield prediction, and the use of RSM for yield improvement or design centering is developed.

Statistical simulation using MSTAT [8][9] which is based on the Response Surface Methodology (RSM)[1][6] provides an effective methodology for yield prediction. Through the use of experimental designs and regression analysis, a polynomial is derived that models circuit behavior with respect to independent process and design parameters. Using this model, six sigma yield estimations are accurately obtainable within a short period of time. By predicting yield, the ability to improve it now is available.

The development of techniques to improve yield for integrated circuits has been restricted to the estimation or approximation of the region of acceptability (R_A). This region of acceptability is critical for the estimation and improvement (design centering) of yield. To improve the accuracy of the design centering process the definition of the region of acceptability has to improve. The importance of accurately defining the region of acceptability is posed by Spence [8], "If the specification were, somehow, to be transformed into component parameter space, the results would be to define a region known as the region of acceptability (R_A)."

This illusive area has been the target of significant studies [4][7][10]. Each of these studies however were not able to accurately define the region of acceptability because of the estimation or approximation techniques that were employed. The limited accuracy of these approaches were due to the confinement of a specific area type, such as a convex hull [4][7] or the estimation of the region of acceptability by statistical exploration techniques[10].

However, with the introduction of MSTAT to model the process variation and to predict yield [1][8][9], a new method has been developed that can accuracy define the region of acceptability with respect to the specification and the component parameter space by graphically (mathematically) intersecting the specification and the component parameter space. This intersection consists of two multiple dimensional surfaces. In the case of a 3 dimensional surface, the intersection of the response surface (component parameter space) with a plane (specification) defines the region of acceptability (See Figure 4).

This report discusses the use of MSTAT in yield predicting and RSM to provide yield improvement (design centering) by transforming the specification into component parameter space. A case study is included to support the proposed method.

II YIELD PREDICTION USING MSTAT

Current approaches of modeling CMOS and bipolar component (electrical) characteristics are by the traditional sampling of test structures contained in circuit dies or wafers that are passed through a given manufacturing line. Extraction of the electrical parameters are performed, the variations are noted and a distribution assigned to the parameters. These electrical parameters however may not be the same parameters monitored and controlled by manufacturing. To predict the behavior of a circuit, Monte Carlo simulations are performed to model the process variations and establish circuit quality (yield). This technique of statistical simulation is computationally intensive, requires a large number of input parameters and is not suited for process control. To overcome the deficiencies from the extraction of electrical parameters, a technique that describes electrical characteristics using physical or process parameters has been developed [3].

Modeling of the electrical characteristics by using independent process parameters is accomplished by the definition of a function that contains the following:

$$P_{ei} = f(P_{pn},P_{pn}) \qquad (1)$$

Where P_{ei} is the electrical parameters and P_{p1} through P_{pn} are the independent processes parameters. Using the Gummel-Poon model parameter BF, the Functional Model Parameters is:

$$BF = K_0 * \rho sbe \qquad (2)$$

The independent process parameters are extracted at wafer probe in order to establish a statistical database. In providing these independent process parameters along with the geometric description (length of the emitter) an accurate statistical model of the electrical component is provided.

The implementation of the Functional Model Parameters is accomplished through the use of a geometric based circuit simulation model[5]. This model consists of geometric and other layout related parameters such as matching, etc. The geometric parameters are defined on the schematic symbol (See Figure 1), values such as the emitter length, emitter width and collector width are described by the parameters LE, WE, WC. Emitter, base and collector stripes are defined using the parameters NE, NB and NC.

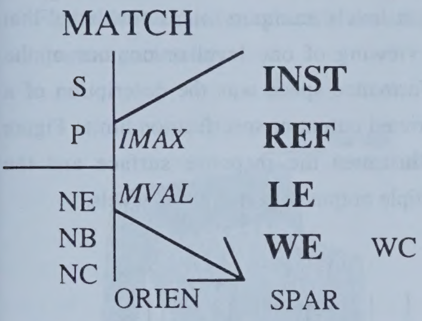


Figure 1. NPN Symbol with Geometric Properties

Additionally layout orientation properties (ORIEN for base/emmitter placement, S for number in series, P for number in parallel) and matching characteristic (MATCH, REF for device reference type) are defined and passed into the functional model parameter equations. Providing these techniques of describing the electrical parameters using independent process parameters, the ability to statistically simulate the characteristics of a parts is now obtainable.

The use of statistical techniques provides the ability to define polynomial models from data collected from circuit simulations. These models can then be used to reduce the time required to simulate the circuits behavior or quality from variations in the manufacturing process. The four statistical simulation/analysis modes that MSTAT supports are:

1)	Sensitivity Analysis	2)	Response Surface Methodology (RSM)
3)	Monte Carlo/Yield Prediction	4)	Cpk

III YIELD IMPROVEMENT (DESIGN CENTERING) USING RSM

The optimization or improvement in yield which is called design centering is the focus of this study. To improve yield it must accurately be defined and it's definition must be such that it can be applied to the design centering process or yield improvement. A comprehensive definition of yield and its application to design centering is defined by Spence[10]. It is; "If the specification were, somehow, to be transformed into component parameter space, the result would be to define a region known as the **region of acceptability** (R_A). Figure 2 graphically defines the region of acceptability.

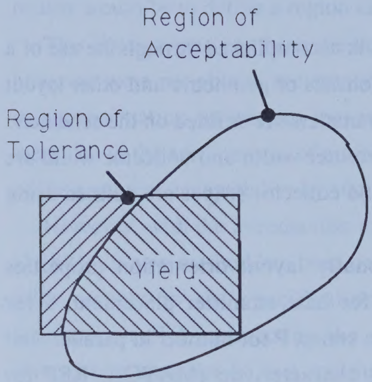


Figure 2. Definition of Region of Acceptability and Region of Tolerance

From the development that was achieved with MSTAT on modeling component behavior using RSM. It was the target of investigation to determine if the R_A as defined by Spence could be identified. Observing that the response surface is the performance space and it is defined by an infinite number of output levels (contours), it is concluded that the viewing of one level or contour of the performance space was the description of a restricted output or specification limit. Figure 3 illustrates the response surface and the multiple output or performance levels.

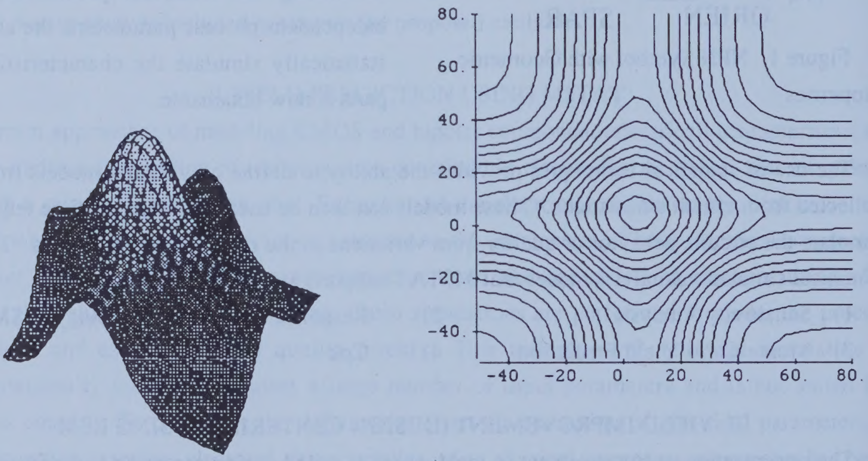


Figure 3. Illustration of Response Surface or Performance Space and Contour Plot or multiple performance levels

The definition of the specification with respect to the response surface is defined by the function:

$$f_{\text{spec}} = f_{\text{poly}} \tag{3}$$

where: f_{spec} is the specification function or limits

f_{poly} is the performance space or polynomial model

Since the solving of function (3) can be computational expensive and in some cases impractical, the expression of (3) can be defined by the following inequality:

$$S_l < RS < S_h \tag{4}$$

Where:

S_l is the lower bound specification

S_h is the upper bound specification

RS is the Response Surface model

Figure 4 illustrates graphically the solution of (3) for a two parameter specification and two component parameters.

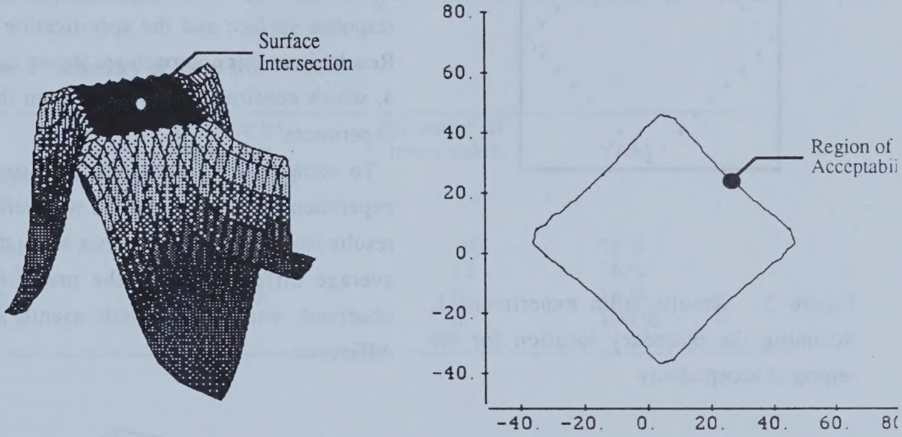


Figure 4. Surface description with surface intersection and region of acceptability defined

Since the response surface or polynomial model has no restrictions, the region of acceptability has no restrictions on boundary types (i.e. convex). For example, in Figure 3 the definition of the region of acceptability can range from a convex type to four separate regions. Each region type is based upon the specification limits with respect to the performance space.

Concluding that this limited output or single level contour is the region of acceptability based on Spence's definition, a series of experiments were performed to verify this hypothesis. The first experiment that was conducted involved the recapturing of points that lie on the perimeter of the region of acceptability using a sequential scanning of the response surface and the solving the equality:

$$900 < X < 950 \tag{5}$$

with X being the response surface of the gaussian function:

$$1/\text{sigma}*\sqrt{2}*\pi*\exp(-(p1-\text{mean})^{\wedge}2/2*\text{sigma})+1/\text{sigma}*\sqrt{2}*\pi*\exp(-(p2-\text{mean})^{\wedge}2/2*\text{sigma}) \quad (6)$$

where: sigma is the sigma level of measurement, p1 & p2 are the independent parameters.

Results from the experiment are described in figure 5.

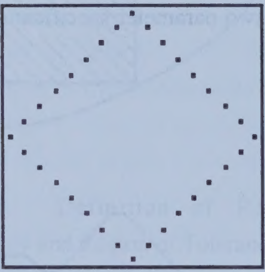


Figure 5. Results from experiment 1. Scanning the boundary location for the region of acceptability

To confirm the results obtained, from experiment one, *Mathmatica* was used to mathematically solve the intersection of the response surface and the specification plane. Results from this approach are shown in figure 4, which confirmed the results from the first experiment.

To compare the results obtained from both experiments, residual analysis was performed results from a residual analysis show that the average difference from the predicted and observed were eight onethousanth or 4% difference.

To further confirm the observation of experiment #1 the application of this new approach in defining the region of acceptability and the maximization of yield was applied to the MC34074 [8] with arbitrarily poor input parameter mean values. For this experiment the Fibanachi average of two NPN base currents are measured. Given these poor values the initial yield 58.824%, was obtained.

An important criteria for design centering with the MC34074 is to provide engineering judgement and experience in the design centering process because of the sensitivity of the input parameters. This sensitivity causes unpredictable results in circuit behavior. To avoid this undesirable region (the lower range) a design engineer must be allowed to use the knowledge gained from engineering experience to determine what is the acceptable performance (region of acceptability) distance from the sensitive lower range. This results in the "best" solution given the circumstances provided.

A hypothetical consideration in the design centering process is that parameter p1 (vpsb) is highly sensitive. Because of this consideration, the design centering process requires the judgement of the designer in determining the "best" solution for yield maximization.

Identifying from the sensitivity analysis that MSTAT[8] produced, the independent parameters p1 and p2 contributed 84.63% of the effect output response. With this high percentage of effect, these two independent parameters were identified to be adjusted for yield improvement, while all other independent parameters were held at their nominal value. Results of this consideration produced the following response model:

$$\begin{aligned} FIBAVG = \; & 0.01034 - 0.00671*p1 - 0.00265*p2 + 0.00175*p1*p1 + \\ & 0.00046*p1*p2 + 0.00057*p2*p2 \end{aligned} \tag{7}$$

Results in yield improvement from adjusting the mean[1] values of p1 and p2 are defined in Table 1.

Table 2 summary of yield improvement

Design Centering Step	Parameter p1 mean value	Parameter p2 mean value	Yield
initial	20	-10	58 %
step 1	-9	12	76 %
step 2	-10	0	88 %
step 3	-7	0	100 %

Figure 6 shows the response surface as defined by (6), Figure 7 illustrates the initial R_A that was determined by mathematical derivation, response surface scanning and a contour plot. The response surface scanning and the contour plot were used to confirm the results of the mathematical derivation. Tables 2 and three document the residual analysis for the initial and final design centering steps.

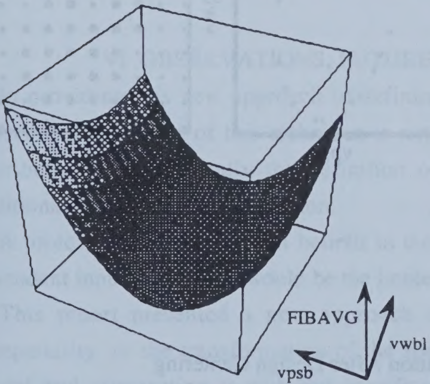


Figure 6. Three dimensional representation of RSM model FIBAVG

---

[1] A similar approach to design centering by the adjusting the mean values of dependent input parameters was conducted by Antriech[2]

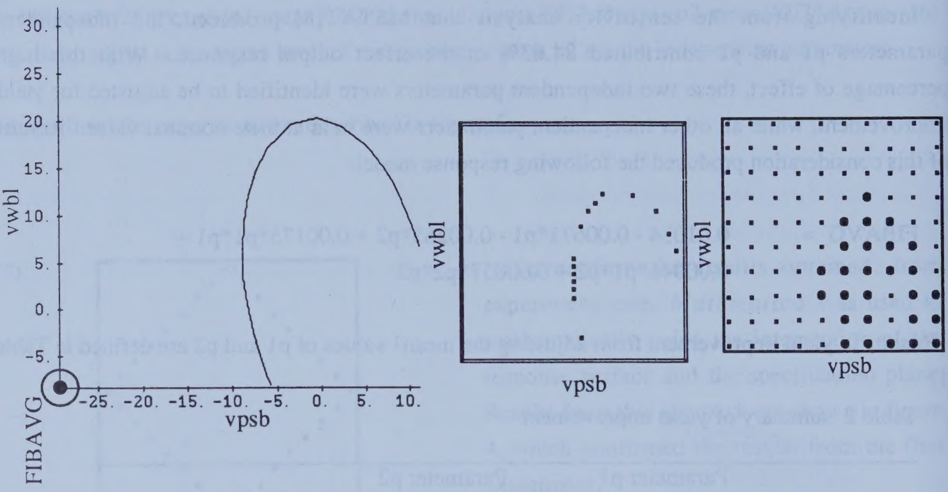


Figure 7. Initial Yield estimation of MC34074

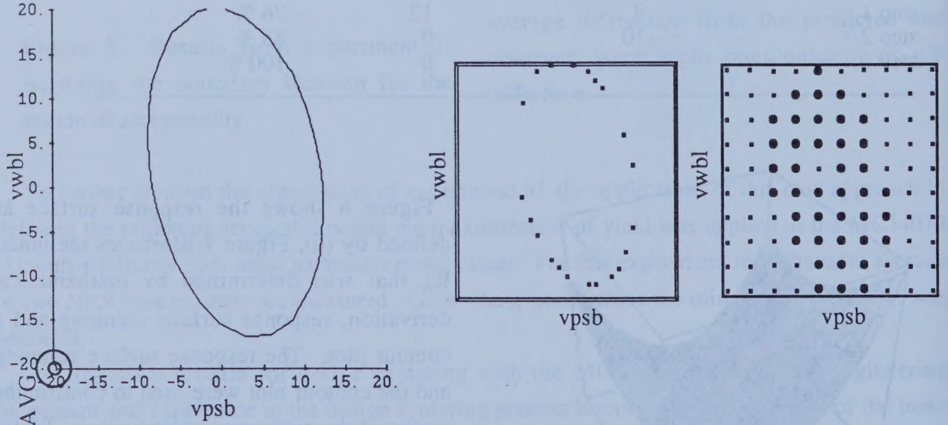


Figure 8. Final Yield Estimation After Design Centering

Table 2

Residual analysis results from initial design centering step

vpsb	vwbl	FIBAVG(Y)	Y-Y(avg)	vpsb'	vwbl'	FIBAVG'(Y')	Y'-Y'(avg)	Residual (Y-Y')
1.46	-16.3	0.187965	-0.00085	1.5	-17	0.191735	-0.00344	-0.00377008
8.88	-14.2	0.183311	-0.00551	9	-15	0.1976	0.002427	-0.01428896
10.1	9.03	0.185589	-0.00323	10.5	9	0.198613	0.00344	-0.013024307
12.1	-7.5	0.195559	0.006742	12	-7.5	0.192357	-0.00282	0.003202
11.9	2.96	0.191662	0.002845	12	3	0.19556	0.000387	-0.003898348
	Average	0.188817			Average	0.195173	Average	-0.006355939

Table 3

Residual analysis results from final design centering step

vpsb	vwbl	FIBAVG (Y)	Y-Y(avg)	vpsb'	vwbl'	FIBAVG'(Y')	Y'-Y'(avg)	Residual (Y-Y')
-7.88	-1.99	0.186624	-0.00686	-8	-2	0.19096	-0.00453	-0.004335891
-8.04	12.6	0.187915	-0.00557	-8	14	0.19912	0.003634	-0.01120544
-5.94	-7.97	0.191048	-0.00243	-6	-8	0.19336	-0.00213	-0.002311659
-2.05	-13.6	0.185742	-0.00774	-2	-14	0.19246	-0.00303	-0.006718125
0.049	-16	0.197973	0.004490	-1	6	0.19866	0.003174	-0.000686618
1.99	19.4	0.184791	-0.00869	2	20	0.19732	0.001834	-0.012528765
7.98	-15.4	0.187696	-0.00579	8	-16	0.1981	0.002614	-0.01040422
7.98	13.7	0.189203	-0.00428	8	14	0.1948	-0.00069	-0.00559684
9.76	9.99	0.186815	-0.00667	10	10	0.19474	-0.00075	-0.007925139
11.9	7.97	0.237023	0.04354	12	-8	0.19534	-0.00015	0.041682693
	Average	0.193483			Average	0.195486	Average	-0.002003

VI OBSERVATIONS, FUTURE RESEARCH AND CONCLUSION

In providing this new approach to defining the region of acceptability using RSM it was observed that the use of this technique is not restricted to parameters defined using gaussian distributions. Additionally the definition of the region of tolerance does not require the traditional linear boundary definition.

A topic of research that will benefit in the observation of the design centering process for dependent input parameters would be the implementation of the Yield Prediction Formula.

This report presented a new approach in design centering by defining the region of acceptability or the transformation of the specification into parameter space by using a RSM model and intersecting it with the specification limits or surface. In support of this new technique experimental results were presented, along with a case study using the MC34074 was used to demonstrate that improvement in yield that can be achieved.

REFERENCES

[1] Alvarez, A. Statistical Design and Response Surface Methods to Computer-Aided VLSI Device Design," IEEE Trans. on Computer-Aided Design, 7, No. 2, (February 1988), 272-288.

[2] Antreich, K. J. and Koblitz, R. K., "An interactive procedure to design centering," Proceedings IEEE International Symposium on Circuits and Systems, 1981, 139-142.

[3] Davis, W. F. and Ida, R. T. "Statistical IC Simulation Based on Independent Wafer Extracted Process Parameters and Experimental Designs," IEEE Proc of Bipolar Circuits and Tech Meeting (September 1989), 262-265.

[4] Director, S. W. and Hachtel, G. D., "The simplicial Approximation Approach to Design Centering," IEEE Trans on Circuits and Systems, CAS-24, No. 7 (July 1977), 363-371.

[5] R. Hester, etal., "A Process and Geometric Driven Device Macro Model for Statistical Simulation of Bipolar IC's," *IEEE* ISCAS-89.

[6] Khuri, A. I., Response surfaces: design and analyses (New York: M. Dekker, 1987).

[7] Low, K.K. and Director, S.W., "A new Methodology for the Design Centering of IC Fabrication Processes" IEEE Proc ICCAD-89 (1989), 189-201.

[8] Rencher, M.R. and Salamina, N., "Statistical Bipolar Circuit Simulation Using MSTAT," IEEE Proc ICCAD-89 (1989), 189-201.

[9] Rencher, M.R., "Analog Statistical Simulation", IEEE Proc. CICC-1991, pp 29.2.1-29.2.4.

[10] Spence, R. and Soin, R. S., Tolerance Design of Electronic Circuits (Wokingham, England: Addison-Wesley, 1988), pp. 14-17, 27-87, 88-103.

Pebble Shift with Prespecified Paths[1]

J. Narasimhan
Elec. Eng. Dept. and
Systems Research Ctr.
Univ. of Maryland
College Park, MD 20742

C. S. Rim
Dept. of Comp Sci.
Sogang Univ.
Seoul
Republic of Korea

K. Nakajima
Elec. Eng. Dept. and
Inst. for Adv. Comp. Studies
Univ. of Maryland
College Park, MD 20742

Introduction

In a recent study Kumar *et al.* [2] have proposed a scheme for enhancing the yield of programmable gate arrays (PGAs) that exploits advances in physical restructuring. In their scheme an initial placement of a circuit is first obtained on a nondefective PGA using a technique like simulated annealing. To implement the circuit on a defective PGA with the same architecture, this placement is reconfigured using some very fast method. The problem of reconfiguration, which is central to their yield enhancement scheme, has been modeled in terms of a graph and pebbles by Narasimhan *et al.* [4] and called the *pebble shift problem* (PS).

In their model a vertex is created for each physical gate of a PGA. Vertices representing adjacent gates are connected by an edge of appropriate length. Each vertex in the graph is classified as *defective* or *nondefective* depending upon whether the gate it represents is defective or nondefective, respectively. Each logic gate of a circuit is regarded as a pebble of appropriate integer weight which depends on the number of terminals used by the gate. An initial placement of the circuit on the PGA is seen as a placement of the pebbles on the corresponding graph with at most one pebble on a vertex. Each pebble that is located on a defective vertex, say, d must be moved to a nondefective vertex, by shifting all the pebbles along a path from d to a vacant and nondefective vertex f. A remapping of the logic gates on a chip may be seen as a sequence of such shifts of pebbles on the graph that leaves no pebble on a defective vertex. The problem is to determine a set of paths and their correct order so as to minimize an associated cost of the shifts applied along the paths in the specified order.

This problem has been shown to be NP-hard by Narasimhan *et al.* [3]. In this paper we consider a variation of PS, which we call the *pebble shift sequencing problem* (PSS). In addition to the input required for PS, a set of prespecified paths along which shifting must be performed is given for PSS. The problem is to determine a sequence in which the paths from this set are to be applied for shifting with a minimum associated cost.

Problem Statement

Let $G = (V, E)$ be a graph in which each vertex is classified as defective or nondefective and each edge $(u, v) \in E$ is associated with a length $l(u, v)$. Let P be a set of pebbles such that $|P| \leq |V|$. Initially all the pebbles are placed on the vertices of G with at most one pebble on a vertex. This placement is called the *initial placement* and denoted by C_1. A vertex v is said to be *occupied* if a pebble is placed on it and *vacant* otherwise. In the former case let $p(v)$ denote the pebble placed on v. Let D_1 be the set of occupied and defective vertices in the graph and let $|D_1| = m$.

A path $[d = v_1, v_2, \ldots, v_l = f]$ is called a *reconfiguration path* and denoted by $Q(d, f)$, if d is occupied and defective, f is vacant and nondefective, and for $i = 2, \ldots, l-1$, v_i

---

[1] This work was supported in part by National Science Foundation Grants MIP-84-51510 and CDR-88-03012 and in part by a grant Nissan Motor Co., Ltd.

is either vacant and defective, or occupied. For such a reconfiguration path $Q(d,f)$, let $u_1, u_2, \ldots, u_r$ be the occupied vertices that appear in this order along the path and let $u_0 = d$ and $u_{r+1} = f$. An operation called the *reconfiguration* along $Q(d,f)$ is defined to be the successive shifting of each pebble $p(u_i)$ from u_i to u_{i+1} for $i = r, r-1, \ldots, 0$. A *reconfiguration sequence* R is a sequence of reconfiguration paths $Q(d_i, f_i)$ for $i = 1, 2, \ldots, m$, which when applied successively to the C_1 will result in a final placement C_{m+1} with no occupied and defective vertex.

We define four cost measures for PSS. Let $v^1(p)$ and $v^{m+1}(p)$ denote the vertices on which a pebble p is located in the placements C_1 and C_{m+1}, respectively and let $l^*(v^1(p), v^{m+1}(p))$ denote the shortest distance between $v^1(p)$ and $v^{m+1}(p)$. The *total displacement* due to the application of a reconfiguration sequence R is defined as $\sum_{p \in P} w(p) \times l^*(v^1(p), v^{m+1}(p))$. The *maximum displacement* of R is defined as $\max_{p \in P}\{w(p) \times l^*(v^1(p), v^{m+1}(p))\}$. For a reconfiguration path $Q(d,f)$ with occupied vertices $u_1, u_2, \ldots, u_r$ as defined earlier, let z_i^1, $z_i^2 \ldots$, $z_i^{i_k}$ be the vacant and defective vertices, if any, that appear in this order between $u_i = z_i^0$ and $u_{i+1} = z_i^{i_k+1}$. The *weighted shift* of each pebble $p(u_i)$ due to this reconfiguration, denoted by $sp(p(u_i), Q(d,f))$, is given by $w(p(u_i)) \times a(u_i, u_{i+1})$, where $a(u_i, u_{i+1}) = \sum_{r=0}^{k} l(z_i^r, z_i^{r+1})$. The weighted shift due to a reconfiguration along $Q(d,f)$ is given by $\sum_{i=1}^{r} sp(p(u_i), Q(d,f))$ and is denoted by $sq(Q(d,f))$. The *total shift* due to the application of R, is defined as $\sum_{Q(d,f) \in R} sq(Q(d,f))$. The *maximum shift* of R is defined as $\max_{p \in P}\{\sum_{Q(d,f) \in R} sp(p, Q(d,f))\}$. In the following definition of the problem $Cost(R)$ could be any one of the above four costs to be minimized.

The Pebble Shift Sequencing Problem (PSS)

Instance: A graph $G = (V, E)$, a set of pebbles P, an initial placement of the pebbles on G with a set D_1 of occupied and defective vertices, a set F_1 of vacant and nondefective vertices, a set S of reconfiguration paths each starting at a distinct vertex $d \in D_1$ and ending at a distinct vertex $f \in F_1$, and an integer c.

Question: Is there a sequence R of the paths in S with $Cost(R) \leq c$?

Results

We have proved the NP-completeness of PSS for undirected and directed acyclic cubic planar graphs and for undirected and directed acyclic grid graphs for each of the four cost functions, even when the weight of each pebble is 1 or 2. Our proofs are based on transformations from the planar version of the 'exact cover by three sets' (X3C) problem which has been proved by Dyer and Frieze [1] to be NP-complete even when each element is contained in two or three subsets.

References

1. M. E. Dyer and A. M. Frieze, "Planar 3DM is NP-complete," *Journal of Algorithms*, Vol. 7, 1986, pp. 174–184.
2. V. P. Kumar, A. T. Dahbura, F. H. Fischer, and P. M. Joula, "An Approach for the Yield Enhancement of Gate Arrays," *Proc. IEEE Int. Conf. on Computer-Aided Design*, Santa Clara, CA, Nov. 1989, pp. 226–229.
3. J. Narasimhan, C. S. Rim, K. Nakajima, and A. T. Dahbura, "Shifting Pebbles on a Graph," presented at the 24th Annual Conference on Information Sciences and Systems, Princeton, NJ, March 1990.
4. J. Narasimhan, C. S. Rim, K. Nakajima, and A. T. Dahbura, "Yield Enhancement of Programmable Chips by Reconfiguration," *Proc. of the IEEE Int. Conf. on Wafer Scale Integration*, San Francisco, CA, Jan. 1991, pp. 178–184.

Improved Global Routing using λ-Geometry *

Surendra Burman, Hang Chen, Naveed Sherwani
Department of Computer Science
Western Michigan University
Kalamazoo, MI 49008
sherwani@cs.wmich.edu

Abstract

We study the problem of global routing of multi-terminal nets in a generalized geometry called λ-geometry in order to improve layout and consequently enhance the performance. In λ-geometry, edges with angles $i\pi/\lambda$, for all i, are allowed, where λ (≥ 2) is a positive integer. $\lambda = 2$, 4 and ∞ correspond to rectilinear, 45° and Euclidean geometries respectively.

In this paper, we prove that all minimum spanning trees in λ-geometry satisfy the *separability* property, for $\lambda = 4$. Furthermore, we extend our result and prove that all minimum spanning trees are separable for all λ-geometries, $\lambda > 4$. We develop a polynomial time algorithm to find an optimal steiner tree in λ-geometry which is derivable from the minimum spanning tree. Our experiments indicate that tree length can be reduced upto 10-12% by using 4-geometry as compared to rectilinear goemetry, however length reduction is quite marginal for higher geometries. As a consequence, we conclude that for purposes of global routing it is sufficient to consider layouts in 4-geometry.

1 Introduction

Global routing of multi-terminal nets is an important problem in physical layout and design of VLSI circuits. This problem has been traditionally viewed as a steiner tree problem [1, 8]. Due to this application and many other uses, steiner trees have been a subject of intensive research [1, 2, 4, 6, 5, 7, 8, 9, 11]. A tree interconnecting a set $P = \{p_1, p_2, \ldots, p_n\}$ of specified points (*demand points*) in a cartesian plane and some arbitrary points $Q = \{q_1, q_2, \ldots, q_m\}$ (*steiner points*) is called a steiner tree of P. The location and number of steiner points is arbitrary. A steiner tree with minimum total length is called a Steiner Minimum Tree (SMT). It has been shown that finding a Steiner Minimum Tree is NP-complete [2]. In view of the NP-completeness of the problem, several heuristic algorithms have been developed for this problem.

The steiner tree whose edges are constrained to be of rectilinear shape is called *Rectilinear Steiner Tree* (RST). It is well known that any minimum cost rectilinear spanning tree (RMST) is a $\frac{3}{2}$ approximation of a SMT [5]. As a result, many heuristic algorithms use minimum spanning tree (MST) as a starting point and apply local modifications to

---

*Research supported in part by National Science Foundation under grant USE-9052346

obtain a SMT. In this way, these algorithms can guarantee that weight of the SMT is at most $\frac{3}{2}$ of the weight of the optimal tree [3, 4, 8, 9]. In [8], the concept of separability was introduced. Two non-adjacent edges are called *separable* if their corresponding minimum enclosing rectangles do not overlap. A tree is called separable if all the pairs of non-adjacent edges are separable. Based on this idea, it was shown that it is possible to develop a polynomial time algorithm to find optimal steiner tree derivable from a separable minimum spanning tree.

The restriction of layout to rectilinear geometry and thus restriction to the use of only rectilinear steiner trees was necessary to account for restricted computing capabilities. Recently, due to enhanced computing capabilities and need for high performance layout designs, non-rectilinear geometry is gaining ground. In order to obtain smaller length steiner trees, we have recently introduced the concept of λ-geometry [10]. In *λ-geometry* only edges with angles $i\pi/\lambda$, for any integer i, are allowed, where λ = 2,4,6,8....∞. Obviously we can see that λ-geometry always includes rectilinear edges and is a useful goemetry with respect to the fabrication technologies.

In this paper, we study the concept of separability in λ-geometry. We show that all minimum spanning trees in λ-geometry satisfy the *separability* property, for $\lambda = 4$. Furthermore, we extend our result and prove that all minimum spanning trees are separable for all λ-geometries, $\lambda > 4$. We extend the algorithm presented in [8] to λ-geometry. Thus our algorithm finds an optimal steiner tree in λ-geometry which is derivable from the minimum spanning tree. Our experiments indicate that tree length can be reduced upto 10-12% by using 4-geometry as compared to rectilinear goemetry, however length reduction is quite marginal for higher geometries. As a consequence, we conclude that for purposes of global routing it is sufficient to consider layouts in 4-geometry.

The rest of this paper is organized as follows. Section 2 discusses some basic concepts about λ-geometry. In section 3, we establish our main result related to λ-separability of minimum spanning trees. Section 4 outlines an algorithm for optimal steiner tree in λ-geometry and we conclude in Section 5.

2 Basic Concepts of λ-Geometry

In this section we establish some basic properties of λ-geometry which will be used in later sections of the paper. The concept of λ-geometry is very closely related to uniform geometry [11]. The key difference is inclusion of rectilinear orientations in all λ-geometries.

Definition 1 *A λ-angle is defined as an angle of measure $i\pi/\lambda$ for an integer i ($0 \leq i < \lambda$), where $\lambda = 2k$ for k being a positive integer.*

Definition 2 *A λ-parallelogram is a parallelogram with boundaries at λ-angles.*

Definition 3 *Let $e = p_1p_2$ be an edge. A λ-neighborhood of e is the closure of a λ-parallelogram with e as a diagonal. We denote a λ-neighborhood of e as $N_\lambda(e)$.*

For example, when λ=4, only the six parallelograms shown in Figure 1 are possible λ-neighborhoods. We denote them as type 1 through type 6 as shown in Figure 1. We observe that for a given edge e there exists a λ-neighborhood which is enclosed in all other λ-neighborhoods. Such a λ-neighborhood is called a smallest λ-neighborhood of e.

For a given edge with the angle of measurement of 0° and 45°, the type 1 λ-neighborhood is the smallest. It is easy to verify that this fact holds for type 2 (45° − 90°), type 3

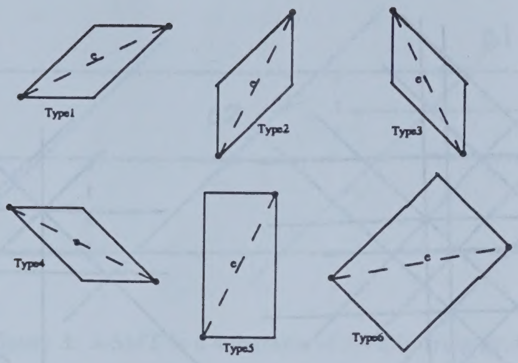


Figure 1: Types of Parallelograms in 4-neighborhood

(90° – 135°) and type 4 (135° – 180°) as well. In fact, only type 1, type 2, type 3 and type 4 λ-neighborhoods are possible to be smallest. We state this observation as the following lemma.

Lemma 1 *For a given edge e, there exists a smallest 4-neighborhood which is among type 1, type 2, type 3 and type 4.*

Definition 4 *A λ-layout of an edge $e = p_1p_2$ is the path from p_1 to p_2 along the smallest λ-neighborhood of e.*

The number of λ-layouts of an MST edge is an exponential function of the number of points on the subgrid within the bounding parallelogram of the two end points of the edge. Choosing different λ-layouts for the edges of the MST will result in different amount of overlaps among these layouts. If we merge these overlapping portions by introducing appropriate steiner points, the resulting structure is a steiner tree of the given point set. These steiner points are a result of merging the overlaps between the various λ-layouts.

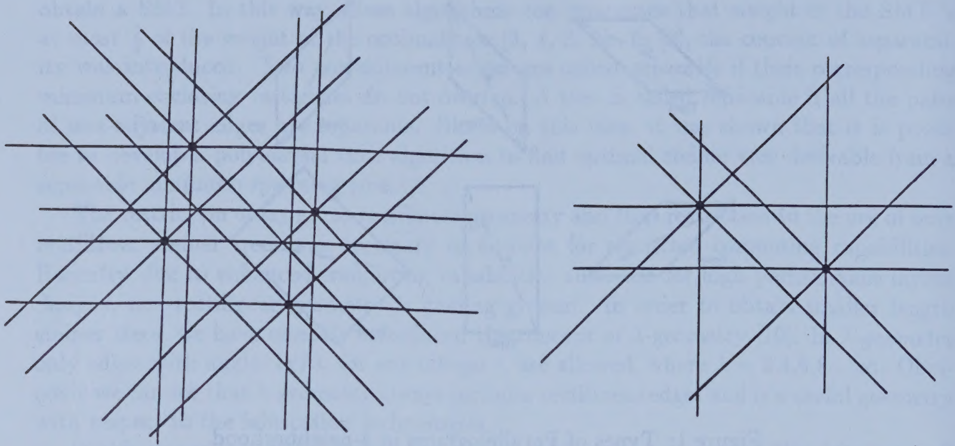
Definition 5 *Let $e = p_1p_2$ be an edge. The λ-distance between p_1 and p_2, or λ-length of e, is defined as the length of the λ-layout of e. We denote λ-distance as $d_\lambda(p_1, p_2)$.*

Definition 6 *A λ-MST (λ-minimum spanning tree) of a point set K in the Euclidean plane is a minimum spanning tree, of the complete graph $G(K)$ over K, with respect to λ-distance.*

Definition 7 *A λ-ST (λ-steiner tree) is a steiner tree in which all edges are at λ-angles.*

In considering λ-Steiner Minimum Tree (λ-SMT), we shall first identify the complete set of possible steiner points. The nodes of λ-SMT which are not demand points and have degree at least 3 are the steiner points.

Definition 8 *The λ-underlying grid of a point set K is the grid obtained by drawing the lines through each point of K at all λ-angles.*



(a) Grid Graph for n=4 and λ=4 (b) grid Graph for n=2 and λ=4

Figure 2: Examples of grid graphs in λ-geometry

The following theorem gives a relationship between a given point set K and λ.

Theorem 1 *Let K be a point set in the plane. Then the maximum number of points in the λ-underlying grid is given by $p = \frac{\lambda n(\lambda-1)(n-1)}{2} + n$, where $n = \mid K \mid$.*

Proof: A grid is determined by λ and K. At most there are λn lines in a grid. Therefore there are at most $(\lambda - 1)(n - 1)$ possible steiner points on each line. In this way, we count each possible steiner points on two lines. Therefore, total number of possible steiner points is $\frac{\lambda n(\lambda-1)(n-1)}{2}$. The total number of demand points is given to be n. Hence total number of grid points is $p = \frac{\lambda n(\lambda-1)(n-1)}{2} + n$ ∎

Clearly for rectilinear geometry (that is $\lambda = 2$), the maximum number of points is given by $p = n^2$. Also we see that for Euclidean geometry (that is $\lambda = \infty$), then the number of grid points is infinite as expected. Some examples of grid graphs in λ-geometry are shown in Figure 2.

The optimal 2-SMT of a point set is known to be subgraph of its underlying grid [6]. Therefore it is sufficient to consider routes for the edges of Minimum Spanning Tree on the underlying grid. We extend this result and show that it holds for λ-goemetry as well.

Theorem 2 *There exists a λ-SMT for a given point set on its λ-underlying grid.*

Proof: For sake of brevity, we will only outline the proof. By contradiction assume that we have a λ-SMT which consists of some segments which do not lie on the λ-underlying grid. Consider the layout given in Figure 3. The dotted lines indicate that these line segments do not lie on the λ-underlying grid. Let us consider segment l_1. It can be easily seen that l_1 lies in a λ-parallelogram and we can easily replace l_1 with an equivalent segment on the λ-underlying grid. Since this operation can be repeated for all 'illegal' segments, we can obtain another layout, which has the same cost and consists of only

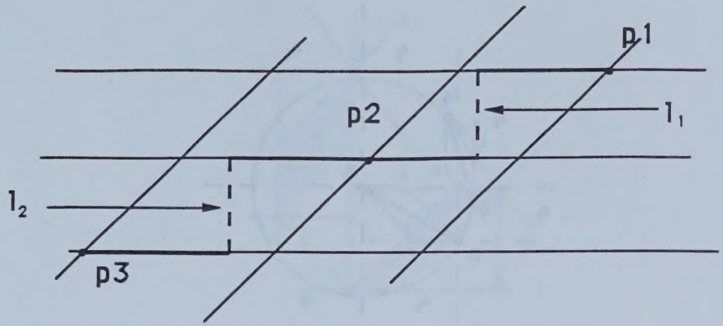


Figure 3: λ-SMT is a subgraph of λ-underlying grid

segments on the λ-underlying grid. This means that a λ-SMT can be constructed using the λ-underlying grid for the given point set. ∎

3 λ-Separability of Minimum Spanning Trees

In this section we establish that any minimum cost spanning tree in 4-geometry is λ-separable. We extend this to show that our result holds for all higher geometries. We start by first defining the concept of λ-separability.

Definition 9 *Let T be a tree. Let e_1 and e_2 be two non-adjacent edges in T. If there exists two λ-neighborhoods $N_\lambda(e_2)$ and $N_\lambda(e_2)$ such that $N_\lambda(e_1) \cap N_\lambda(e_2) = \phi$, then e_1 and e_2 are λ-separable. T is λ-separable if any two non-adjacent edges are λ-separable.*

Theorem 3 *Any minimum spanning tree over a given set in the plane is λ-separable for $\lambda = 4$.*

Proof: Let T be a given minimal spanning tree in 4-geometry. Assume that T is not 4-separable. Then there exists two non-adjacent edges $e_1 = p_1p_2$ and $e_2 = q_1q_2$ such that e_1 and e_2 are not 4-separable. Let $N_4(e_1)$ and $N_4(e_2)$ be two smallest neighborhoods of e_1 and e_2 respectively. Then $N_4(e_1) \cap N_4(e_2) \neq \phi$. By lemma 1, there are four possible smallest neighbors. Since reflections about 90° line or about 45° line do not change the smallest 4-neighborhood.

Without loss of generality, let p_1 be disconnected from e_2 if e_1 is removed from T and let q_1 be disconnected from e_1 if e_2 is removed. We also assume the length of e_1 is greater than or equal to the length of e_2, *i.e.* $|e_1| \geq |e_2|$, where $|e_i|$ denotes the length of the edge e_i, $(i = 1, 2)$.

Let us draw a circle with center at p_1 and $|e_1|$ as radius (See Figure 4). Neither q_1 or q_2 can be inside the circle, otherwise we can remove e_1 and connect p_1 to this point to obtain a smaller cost spanning tree. This contradicts the fact that T is a minimum cost spanning tree. Now, let us assume that both q_1 and q_2 lie on the boundary of the circle. Let us divide the plane into eight regions R_1 through R_8 as shown in Figure 4. The regions, R_1 through R_7 are all considered outside the circle. Under our assumptions that $N_4(e_1) \cap N_4(e_2) \neq \phi$ and $|e_1| \geq |e_2|$, we shall consider positions of q_1 and q_2 in all regions R_1 through R_7.

Case 1 (q_1 and q_2 lie in $R_4 \cup R_5 \cup R_6$): q_1 and q_2 can not both lie in $R_4 \cup R_5 \cup R_6$, otherwise $N_4(e_2) \cap N_4(e_1) = \phi$.

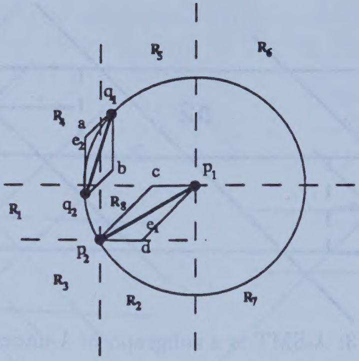


Figure 4: Separability Proof in 4-geometry

Case 2 (q_1 and q_2 lie in $R_6 \cup R_7$): q_1 and q_2 can not both lie in $R_6 \cup R_7$, by the same reasoning as given in case 1, if they do lie in these regions then $N_4(e_2) \cap N_4(e_1) = \phi$.

Case 3 (q_1 and q_2 lie in $R_2 \cup R_3 \cup R_7$): This case is similar to case 1 and 2, q_1 and q_2 can not both lie in $R_2 \cup R_3 \cup R_7$ otherwise $N_4(e_2) \cap N_4(e_1) = \phi$.

Case 4 (q_1 and q_2 lie in $R_3 \cup R_1 \cup R_4$): This case is also easily ruled out because if q_1 and q_2 both lie in $R_3 \cup R_1 \cup R_4$, then by using same argument as in cases above we get $N_4(e_2) \cap N_4(e_1) = \phi$.

Case 5 (q_2 lies at the boundary of R_1 and R_3 and q_1 is in $R_2 \cup R_7$): In this case if $d_4(q_1, p_2) < |e_2|$, then we can remove e_2 and connect q_1 to p_2 to construct a smaller cost spanning tree. This is a contradiction to the fact that T is a minimum spanning tree, otherwise $d_4(q_1, q_2) \geq |e_2|$. The same argument leads to the conclusion that we can not have the neighborhoods of the edges e_1 and e_2 intersect each other.

Case 6 (q_1 lies at the boundary of R_1 and R_3 and q_2 is in $R_2 \cup R_3$): In this case if $d_4(q_2, p_2) < |e_2|$, then we can remove e_2 and connect q_2 to p_2 to construct a smaller cost spanning tree. This is a contradiction to the fact that T is a minimum spanning tree, otherwise $d_4(q_2, q_2) \geq |e_2|$. The same argument leads to the conclusion that we can not have the neighborhoods of the edges e_1 and e_2 intersect each other. In the same way, we can discuss the boundary of R_2 and R_3.

Case 7 (q_2 lies on the boundary of the circle and R_1): In this case since q_2 is on the boundary on the circle, $N_4(e_2) \cap N_4(e_1) = \phi$. We need to consider only the case of q_1 being in region R_5. The boundary case is the worst case as shown in Figure 4. q_1 can not be in R_6 since otherwise $|e_2| > |e_1|$ which is a contradiction. In this case, we know $N_4(e_2)$ is of type 2. Then q_2b is parallel to p_2c. This ensures that $N_4(e_1) \cap N_4(e_2) = \phi$.

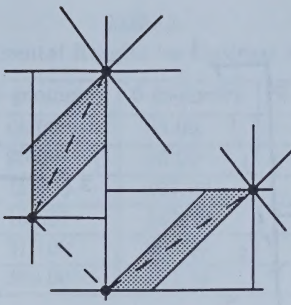


Figure 5: MST Nonseparable in 2-geometry but Separable in 4-geometry

Since we considered all possible cases, we conclude that $N_4(e_1) \cap N_4(e_2) = \phi$, and indeed edges e_1 and e_2 are 4-separable, thus contradicting our assumption. This proves the theorem. ∎

The following theorem establishes a relationship between separability concept in different geometries.

Theorem 4 *If for given a point set a λ-minimum spanning tree T is λ_1-separable then T is also λ_2-separable for $2 \leq \lambda_1 < \lambda_2 \leq \infty$ where λ_1 and λ_2 are even positive integers.*

Proof: It is easy to see that enclosing parallelogram of points p_1 and p_2 in λ_2-geometry properly lies within the enclosing parallelogram of the same points in λ_1-geometry. This is due to the fact that smaller legal angles are allowed in λ_2-geometry as compared to legal angles in λ_1-geometry. It follows from the separability definition that if two edges have non-intersecting parallelograms in λ_1-geometry, then their respective parallelograms in λ_2 geometry will be non-intersecting. ∎

It can be seen from Figure 5 that the converse of the theorem 4 is not true. Theorem 3 and 4 lead to the following result.

Theorem 5 *Any minimum spanning tree for a given point set in the plane is λ-separable for any even $\lambda \geq 4$.*

As a result of this theorem, all minimum spanning trees are separable for $\lambda \geq 4$ and we do not have to construct the separable MST's as needed in the case of 2-geometry. Based on this result, we develop an algorithm for finding optimal steiner tree derivable from the MST in the λ-geometry.

4 Algorithm for Optimal Steiner Tree in λ-geometry

As mentioned above, the concept of separability in rectilinear geometry has been used in [8] to develop an algorithm for optimal steiner tree derivable from minimum spanning tree. The basic idea of this algorithm is to first find separable MST in rectilinear geometry

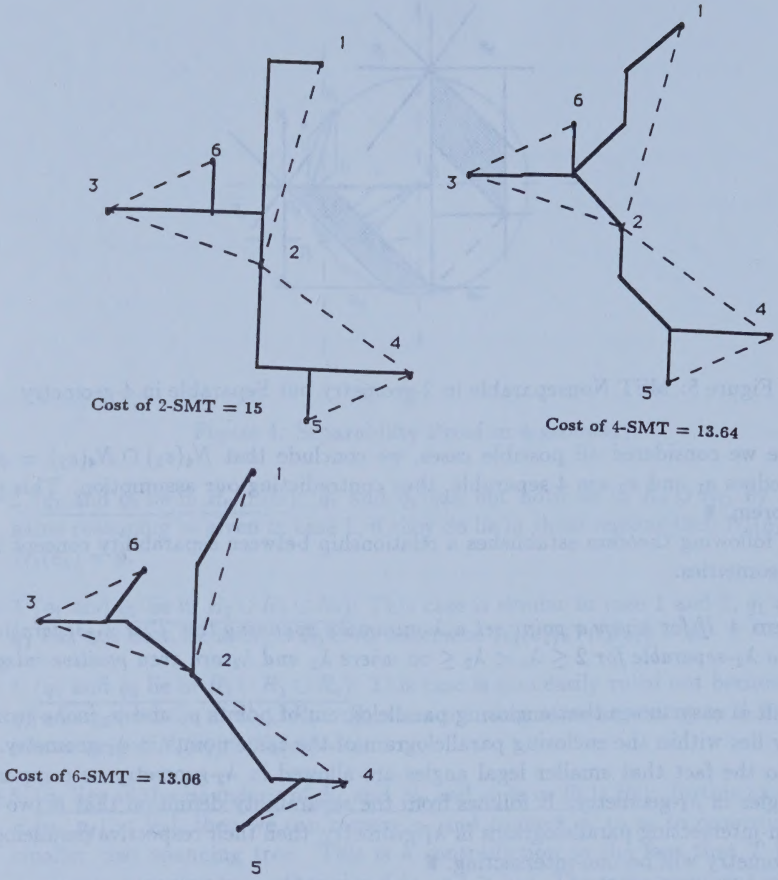


Figure 6: Example of a point set

for the given point set, then construct optimal SMT considering the rectilinear layouts for each edge of the MST and maximizing the overlap between the adjacent edges.

We have developed the algorithm to find optimal steiner trees in λ-geometry. Our algorithm is a generalized version of the one presented in [8] and finds the optimal steiner tree in λ-geometry for any value of λ, details of the algorithm are being omitted for the sake brevity, however a complete description of the algorithm can be found in [10]. We have implemented the algorithm in C on SPARC station 1+. To illustrate our algorithm, let us consider point set in Figure 6. The algorithm presented in [8] produced rectilinear steiner tree with a cost 15.00. Using our algorithm we obtained 4-SMT with cost 13.64 and 6-SMT with cost 13.06. Table I shows the experimental results obtained with several randomly generated examples. It can be observed that average length of the steiner tree can be reduced by 8.72% by using 4-geometry as compared to rectilinear geometry. However, the length is reduced by only 4.18% if 6-geometry is used as compared to 4-geometry. As a result, we conclude that for all practical purposes use of 4-geometry is sufficient for global routing because higher geometries do not significantly contribute to the reduction in length.

Table 1
Experimental Results for Optimal λ-SMT

n	2-geometry	4-geometry	6-geometry	% imp. 2-4	% imp. 4-6
16	75.00	66.00	63.00	12.00	4.55
32	99.00	90.00	86.00	9.09	4.44
64	140.00	112.00	107.00	12.86	4.46
128	169.00	156.57	152.39	7.36	2.67
256	236.00	216.00	208.47	8.47	3.67
512	317.00	309.00	305.56	2.52	1.11

5 Conclusion

In this paper we have studied the problem of global routing of multiterminal nets in a generalized geometry, called λ-geometry in order to reduce the delay and consequently enhance the performance. We have proved that any minimum cost spanning tree in 4 or higher geometry is separable. We have also developed an algorithm for optimal steiner tree in λ-geometry derivable from minimum spanning trees. We have experimentally shown that 4-geometry is sufficient and there is no need to consider geometries beyond 4-geometry for global routing, where the computational aspects become too complex and the gain is not considered worth the efforts. This algorithm can be used by the advanced VLSI design tools in order to reduce the routing length and hence delay.

6 Acknowledgement

We wish to thank Majid Sarrafzadeh for several discussions about this work.

References

[1] C. Chiang, M. Sarrafzadeh, and C. K. Wong, "A Powerful Global Router: Based on Steiner Min-Max Trees", *Proceedings of IEEE International Conference on Computer-Aided Design*, November 7-10, 1989, pp. 2-5.

[2] M. R. Garey and D. S. Johnson, "The Rectilinear Steiner Tree Problem is NP-Complete", *SIAM J. Appl. Math.,32,* 1977, pp. 826-834.

[3] F. K. Hwang, "An O($n \log n$) Algorithm for Rectilinear Minimum Spanning Tree", *Journal of the Association for Computing Machinery*, Vol. 26, No. 2, April 1979, pp. 177-182.

[4] F. K. Hwang, "An O($n \log n$) Algorithm for Suboptimal Rectilinear Steiner Trees", *Transactions on Circuits and Systems*, Vol. 26, No. 1, January 1979, pp. 75-77.

[5] F. K. Hwang, "On Steiner Minimal Trees With Rectilinear Distance", *SIAM Journal of Applied Mathematics*, Vol. 30, No. 1, January 1976, pp. 104-114

[6] M. Hanan, "On Steiner's problem With Rectilinear Distance", *SIAM Journal of Applied Mathematics*, Vol. 30, No. 1, January 1976, pp. 104-114.

[7] J. M. Ho, G. Vijayan, and C. K. Wong, “Constructing the Optimal Rectilinear Steiner Tree Derivable From a Minimum Spanning Tree”, *Proceedings of IEEE International Conference on Computer-Aided Design*, November 7-10, 1989, pp. 5-8.

[8] J. M. Ho, G. Vijayan, and C. K. Wong, “A New Approach to the Rectilinear Steiner tree Problem”, *IEEE Transactions on Computer-Aided Design,* Vol. 9, No. 2, February 1985, pp. 185-193.

[9] D. T. Lee, J. M. Smith, and J. S. Liebman, “An $O(n \log n)$ Heuristic Algorithm Rectilinear Steiner Tree Problem”, *Engineering Optimization*, Vol. 4, No. 4, 1980, pp. 179-192.

[10] S. Burman, N. A. Sherwani “Optimal Steiner Trees Algorithm For 45° and Generalized Grids”, *Technical Report, TR/91-15,* Western Michigan University, Kalamazoo, MI.

[11] M. Sarrafzadeh, and C. K. Wong, “Hierarchical Steiner Tree Construction in Uniform Orientations”, manuscript, 1990.

[12] P. Widmayer, Y. F. Wu, and C. K. Wong, “On Some Distance Problems in Fixed Orientations”, *SIAM Journal of Computing,* Vol. 16 No. 4, August 1987, pp. 728-746.

Design of Multiple Bus Interconnection Networks for Fan-in Computations

R. Vaidyanathan

Louisiana State University, Baton Rouge.

Abstract

A fan-in computation is one, that can be represented as a binary tree and in which the computation proceeds from the leaves to the root. Solutions to problems like maximum/minimum, parity and polynomial evaluation fall in this class of computations. We present a method for constructing multibus networks with 2^p processors and 2^b buses ($2 \leq b < p$) that solve fan-in computations optimally. The number of ports per processor is restricted to two and the number of processors connected to each bus is at most $2^{p-b+1} + b - 2$ (which is very close to the best possible value for $p \gg b$). Our networks are highly scalable and in no case need more than 19 % of the existing connections to be changed (when the network is scaled up); in some cases no change is required.

1 Introduction

One of the key aspects of using multiple processors to solve a problem is the manner in which the processors can communicate with each other. Crossbar networks are fast but costly and very often impractical. On the other hand, a unibus interconnection has a low cost, is simple to implement and expand and is fairly standardized. However, its communication bandwidth is very low.

An intermediate solution is to use multiple buses. Multiple bus-based interconnection networks (*multibus networks*) have been studied extensively with respect to their design, reliability, performance and modeling (see for example [3, 4, 5, 6, 7, 8, 10, 11, 12]). In general, because of the variety of computational problems and the different communication patterns their solutions demand, designing a low cost general purpose multibus network that permits all communication patterns to be mapped efficiently on it, is probably impossible. We therefore restrict our attention to a class of computational problems (henceforth referred to as fan-in computations), whose solutions can be represented as a binary fan-in tree (see for example [1, 2, 9]), in which each level involves computations that take the same amount of time. This class includes a number of important problems like maximum/minimum, parity and polynomial evaluation.

We present a method for designing a 2^p processor, 2^b bus network that solves fan-in problems optimally. Moreover, the proposed network has low cost, the number of input/output ports needed per processor is at most two and the number of processors connected to each bus is at most $2^{p-b+1} + b - 2$. The network also permits an existing configuration to be scaled up with very little change to it.

2 Preliminaries

We denote by $B(p, b)$ a multibus network with 2^b buses and 2^p processors. The design of this network involves specifying which processors are connected to which bus(es). Thus, $B(p, b)$ could be viewed as a $2^p \times 2^b$ matrix of 0's and 1's. The element in row i and column j of $B(p, b)$ will be denoted by (i, j). Processor i is connected to bus j iff $(i, j) = 1$.

We assume that the processors operate synchronously and that each processor has its own local memory. In addition, each processor has some input/output ports whose contents can be written on or read from the buses to which they are connected. We will assume that each processor can read from both its ports or write on one of its ports in one unit of time.

A fan-in computation can be represented as a binary tree with $n = 2^k$ leaves that represent the inputs to the computation. The result of the computation appears at the root. Each internal node of the tree represents an internal computation. We assume that all the computations at a given level of of the tree require the same amount of time. In analyzing the performance of our multibus network, we need to consider only the communication steps and therefore, without loss of generality, we will assume that the number of leaves (inputs to the fan-in computation) is 2^p. We will label the edges in a given level of the fan-in tree from 1 to p with the edges between the leaves and their parents labeled 1 and the edges between the root and its children labeled p. We will refer to these labels as the levels of the edges. Since the number of concurrent communications needed at level ℓ ($1 \leq \ell \leq p$) of the tree is $2^{p-\ell}$, we will assume that $b \leq p - 1$, where 2^b buses are available. The number of steps needed to accomplish all the communications at level ℓ is at least $\lceil 2^{p-\ell-b} \rceil$. The minimum number of steps needed for the above fan-in computation is therefore

$$T_{min}(p, b) = \sum_{\ell=1}^{p} \lceil 2^{p-\ell-b} \rceil = \sum_{\ell=1}^{p-b} 2^{p-\ell-b} + \sum_{\ell=p-b+1}^{p} 1 = 2^{p-b} + b - 1$$

For brevity, we will refer to a multibus network $B(p, b)$ that can solve a fan-in computation with 2^p inputs in $T_{min}(p, b)$ steps as an *optimal* network. We note that the time needed for a step at level ℓ could depend on ℓ. However, including this consideration in our discussion will provide no new insight. We will therefore assume that each step needs one unit of time. Nevertheless, our results also hold for fan-in computations where a step at level ℓ needs $f(\ell)$ time, for any function f.

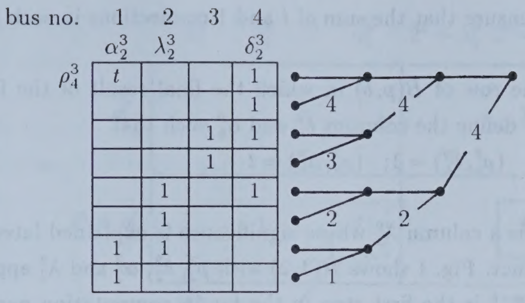


Figure 1: An example of $B(3, 2)$; blank entries are 0

Remarks: Fan-in computations can be solved by connecting $2n - 1$ processors in a binary tree, where n is the number of inputs. However, the number of processors is nearly twice as many as is actually required for an optimal solution. An alternate method would be to map a processor to each path from the leaves to the root. This would form a tree (not a binary tree) of processors. Unlike the proposed multibus network in which each processor has at most two ports, the above processor tree has nodes with non-constant degree, which makes is more difficult to realize in practice.

3 Construction of Optimal Multibus Networks

For $b = 0$ or 1, $B(p, b)$ can be constructed trivially by connecting all the processors to all the buses. This construction though denser than needed (for $b = 1$), still restricts the number of ports per processor to two. We therefore assume that $2 \leq b < p$. Our construction strategy involves three parts.

(*i*) We show that an optimal $B(3, 2)$ exists. This is straightforward and Fig. 1 shows one. Fig. 1 also illustrates the corresponding fan-in tree whose nodes are aligned with the processors (rows) that represent them. Also the communications (edges) are labeled with the corresponding bus numbers used for them. The t entry in Fig. 1 is explained later.

(*ii*) Construction 1: An optimal $B(p + 1, b + 1)$ can be constructed from an optimal $B(p, b)$.

(*iii*) Construction 2: An optimal $B(p + 1, b)$ can be constructed from an optimal $B(p, b)$.

In the context of the matrix $B(p, b)$, we will use the terms row and column synonymously, with processor and bus, respectively. Each entry of $B(p, b)$ can be a 0, 1 or t connection. A 0 or 1 connection indicates whether or not a processor is connected to a bus. A t connection (temporary connection) is used in constructions 1 and 2. A t connection (which is not necessary for the computation) can be replaced by a 0 or 1 connection as required. We will ensure that the sum of t and 1 connections in each row of $B(p,b)$ is at most two.

Definitions: Let the row of $B(p,b)$ in which the final result of the fan-in computation appears be ρ_b^p. We define the columns δ_b^p and α_b^p such that

$$(\rho_b^p, \delta_b^p) = 1; \quad (\rho_b^p, \alpha_b^p) = t \tag{1}$$

In addition, there is a column λ_b^p whose significance is explained later. The columns δ_b^p, α_b^p and λ_b^p are distinct. Fig. 1 shows $B(3,2)$ with ρ_2^3, δ_2^3, α_2^3 and λ_2^3 appropriately labeled.

We note that 2^{p-b} is the first step in the fan-in computation performed by $B(p,b)$, during which some of the 2^b buses are not used. For any bus c of $B(p,b)$, let $\tau_b^p(c)$ be the first step at which c is not used by $B(p,b)$. If c is used during all the steps then $\tau_b^p(c) = T_{min}(p,b) + 1$. For reasons that are explained later, we require $B(p,b)$ to satisfy the following conditions:

$$\left.\begin{array}{ll} (i) & \text{A bus once freed, is never used again.} \\ (ii) & \text{Half the buses used at step } t \geq 2^{p-b} - 1 \text{ are freed at the end of step } t. \\ (iii) & \tau_b^p(\alpha_b^p) = 2^{p-b} \\ (iv) & \tau_b^p(\delta_b^p) = 2^{p-b} + b = T_{min}(p,b) + 1; \quad \text{i.e. } \delta_b^p \text{ is used during all the steps.} \\ (v) & \tau_b^p(\lambda_b^p) = 2^{p-b} + 1 \end{array}\right\} \tag{2}$$

We note that $B(3,2)$ satisfies conditions (1) and (2).

3.1 Construction 1

We start with a $2^{p+1} \times 2^{b+1}$ matrix $\mathcal{M}(p+1, b+1)$ that is divided into four quadrants, each a $2^p \times 2^b$ matrix. The first and third quadrants of $\mathcal{M}(p+1,b+1)$ are $B(p,b)$ and the second and fourth quadrants are all 0's (See Fig. 2). We modify $\mathcal{M}(p+1,b+1)$ to obtain $B(p+1,b+1)$ that has the desired properties. The rows (columns) of $\mathcal{M}(p+1,b+1)$ and $B(p+1,b+1)$ are numbered $1, 2, ..., 2^{p+1}$ (resp. $1, 2, ..., 2^{b+1}$). We replace $(2^p + \rho_b^p, \alpha_b^p)$ (which is a t) by a 0 and $(2^p + \rho_b^p, 2^b + \delta_b^p)$ (which is a 0) by a 1. The resulting $2^{p+1} \times 2^{b+1}$ matrix is $B(p+1, b+1)$.

The fan-in computation with 2^{p+1} leaves can be viewed as a root with two children each of which is a child fan-in computation with 2^p leaves. The first and third quadrants of $B(p+1,b+1)$ can perform the child computations optimally in $T_{min}(p,b)$ time, as they each have 2^b distinct buses. We also note that in constructing $B(p+1,b+1)$ none of the 1 connections of the $B(p,b)$'s have been changed. The partial results of the child computations appear in processors ρ_b^p and $2^p + \rho_b^p$ of $B(p+1,b+1)$. Processor $2^p + \rho_b^p$ sends its partial result to processor ρ_b^p in one step, after which processor ρ_b^p performs the final internal computation. The bus used for the above communication is $2^b + \delta_b^p$. The number of steps needed for the computation is $T_{min}(p,b)+1 = T_{min}(p+1,b+1)$. It is clear from the above discussion that $\rho_{b+1}^{p+1} = \rho_b^p$ and $\delta_{b+1}^{p+1} = 2^b + \delta_b^p$. We assign $\alpha_{b+1}^{p+1} = 2^b + \alpha_b^p$

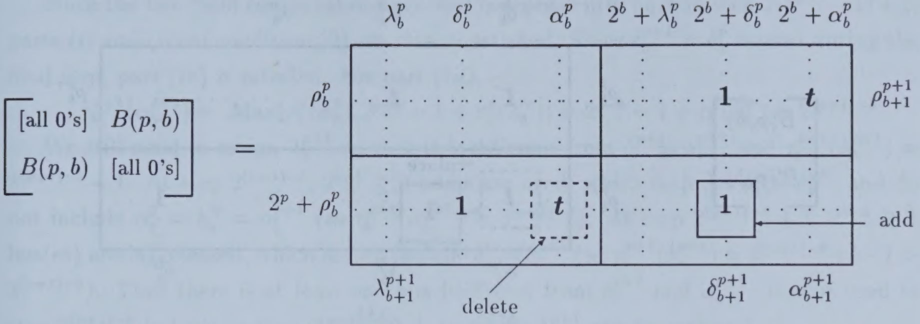


Figure 2: Construction 1

and $\lambda_{b+1}^{p+1} = \lambda_b^p$. It may be verified from Fig. 2 that δ_{b+1}^{p+1}, α_{b+1}^{p+1} and λ_{b+1}^{p+1} are distinct and satisfy condition (1) of section 3.

The only bus used in the last step of the computation performed by $B(p+1, b+1)$ is $\delta_{b+1}^{p+1} = 2^b + \delta_b^p$. Since this was δ_b^p of one of the child computations, it is a bus that has not been freed before. Thus $B(p+1, b+1)$ satisfies part (i) of condition (2). The buses used at the penultimate step are δ_b^p and $2^b + \delta_b^p$. In the final step only $2^b + \delta_b^p$ is used and part (ii) of condition (2) is satisfied. For parts (iii), (iv) and (v),

$$\tau_{b+1}^{p+1}(\alpha_{b+1}^{p+1}) = \tau_{b+1}^{p+1}(2^b + \alpha_b^p) = \tau_b^p(\alpha_b^p) = 2^{p-b} = 2^{(p+1)-(b+1)}$$

$$\tau_{b+1}^{p+1}(\delta_{b+1}^{p+1}) = \tau_{b+1}^{p+1}(2^b + \delta_b^p) = \tau_b^p(\delta_b^p) + 1 = 2^{p-b} + b + 1 = 2^{(p+1)-(b+1)} + (b+1)$$

$$\tau_{b+1}^{p+1}(\lambda_{b+1}^{p+1}) = \tau_b^p(\lambda_b^p) = 2^{p-b} + 1 = 2^{(p+1)-(b+1)} + 1.$$

3.2 Construction 2

Before we proceed, we observe that if $B(p, b)$ is optimal then so is $B'(p, b)$, where $B'(p, b)$ is constructed by permuting the columns of $B(p, b)$. In constructing $B(p+1, b+1)$ from $B(p, b)$ we had a sufficient number of buses available so that the child computations could be scheduled concurrently. All that needed to be done was to get the two partial results to a single processor. For constructing $B(p+1, b)$ from $B(p, b)$, the same scheduling strategy cannot be adopted.

In the first child computation performed by $B(p, b)$ all b buses are used during the first $2^{p-b} - 1$ steps. Up to this point the two child computations can be scheduled one after the other. The additional time (beyond what is needed by one child computation) needed for this is $2^{p-b} - 1$. Beyond this point, each child computation uses at most 2^{b-1} buses (see condition (2), part (ii)) and they can work concurrently. Finally, the two partial results can be brought to a processor in one step. Thus, the above scheduling

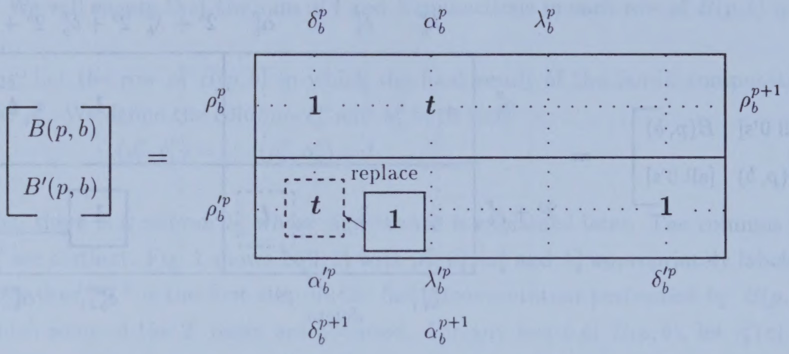


Figure 3: Construction 2

strategy is capable of achieving a time of $T_{min}(p,b)+(2^{p-b}-1)+1 = T_{min}(p+1,b)$. What remains is to show that such a scheduling strategy is indeed possible on $B(p+1,b)$.

Beyond step $2(2^{p-b}-1)$ of the computation performed by $B(p+1,b)$, the two child computations must be able to work concurrently. Therefore, the set of buses used by them (beyond step $2(2^{p-b}-1)$) must be disjoint. This is easy as no more than half the buses are used by each child computation (beyond step $2(2^{p-b}-1)$) and as observed earlier, the columns of $B(p,b)$ may be permuted without affecting its optimality. In addition, there must be at least one bus that connects a pair of processors in the child computations. Otherwise, the final step cannot be done in one unit of time.

We start with a $2^{p+1} \times 2^b$ matrix $\mathcal{N}(p+1,b)$ that is divided into two parts, each a $2^p \times 2^b$ matrix (see Fig. 3). The top half of $\mathcal{N}(p+1,b)$ is $B(p,b)$ and the bottom half is $B(p,b)$ with the columns "suitably permuted". To distinguish the special rows and columns (ρ, α, δ and λ) of $B(p,b)$ from those of $B'(p,b)$ (the bottom half matrix) we will use a " $'$ " (prime) to denote the latter. We map δ_b^p to $\alpha_b'^p$ and α_b^p to $\lambda_b'^p$. At step $2(2^{p-b}-1)+1$, $B(p,b)$ uses 2^{b-1} buses, which includes δ_b^p and does not include α_b^p. The buses released by $B'(p,b)$ at this step include $\alpha_b'^p$ (which is the same as δ_b^p) and does not include $\lambda_b'^p$ (which is the same as α_b^p). Therefore the buses used by $B'(p,b)$ can be mapped on the buses released by $B(p,b)$. Also since the buses once released, are never used later (condition (2) part (i)), the two child computations can proceed independently. We change $(\rho_b'^p, \alpha_b'^p) = (\rho_b'^p, \delta_b^p)$, which is a t connection, to a 1 connection. The resulting $2^{p+1} \times 2^b$ matrix is $B(p+1,b)$. We have not changed the sum of 1's and t's in row $\rho_b'^p$. The results of the child computations appear in processors ρ_b^p and $\rho_b'^p$. Processor $\rho_b'^p$ can now send the partial result to ρ_b^p in one step. The bus $\delta_b^p = \alpha_b'^p$ is used for the final communication.

It is clear that $\rho_b^{p+1} = \rho_b^p$ and $\delta_b^{p+1} = \delta_b^p$. We assign $\alpha_b^{p+1} = \alpha_b^p$. It may be verified from Fig. 3 that δ_b^{p+1} and α_b^{p+1} are distinct and satisfy the condition (1) of section 3.

Since the two child computations proceed independently beyond step $2(2^{p-b}-1)+1$, parts (i) and (ii) of condition (2) are clearly satisfied. Since $\delta_b^{p+1} = \delta_b^p$ is used during the final step, part (iv) is satisfied. For part (iii),

$$\tau_b^{p+1}(\alpha_b^{p+1}) = \text{Max}(\tau_b^p(\alpha_b^p), 2^{p-b} - 1 + \tau_b^p(\lambda_b'^p)) = 2^{p-b} - 1 + \tau_b^p(\lambda_b'^p) = 2^{(p+1)-b}$$

We still need to assign λ_b^{p+1} so that it is different from δ_b^{p+1}, α_b^{p+1} and $\tau_b^{p+1}(\lambda_b^{p+1}) = 2^{(p+1)-b}+1$. At step $2^{(p+1)-b}$, $2^{b-1} \geq 2$ buses are used, which includes $\delta_b^p = \delta_b^{p+1}$ and do not include $\alpha_b^p = \lambda_b'^p = \alpha_b^{p+1}$ (as $\tau_b^{p+1}(\alpha_b^{p+1}) = 2^{(p+1)-b}$). At step $2^{(p+1)-b}+1$, $2^{b-2} \geq 1$ bus(es) are(is) released, which do not include $\delta_b^p = \delta_b^{p+1}$ (as $\tau_b^{p+1}(\delta_b^{p+1}) = 2^{(p+1)-b}+b-1 > 2^{(p+1)-b}$). Thus there is at least one bus (different from δ_b^{p+1} and α_b^{p+1}) that is used in step $2^{(p+1)-b}$ but not in step $2^{(p+1)-b}+1$, to which λ_b^{p+1} can be assigned.

It can easily be shown that the number of processors connected to a bus is at most $2^{p-b+1}+b-2$ for $p > b \geq 0$. For a multibus network with 2^p processors and 2^b buses, in which each processor is connected to two buses, the largest number of processors connected to a bus is at least 2^{p-b+1}. Our figure is close to this for $p \gg b$.

4 Scalability

The primary focus of this section is on the changes to the existing connections that have to be made when the $B(p,b)$ is scaled up to $B(p+r, b+s)$, where $r, s \geq 0$. Since there are two ports per processor, the processors themselves need not be modified for scaling up a configuration. Also for each step in the fan-in computation a bus is accessed by no more than two processors. Therefore, scaling up the system need not involve augmenting the driving capabilities of the buses. The permutation of the buses that is needed for construction 2 is not considered.

In general $B(p+r, b+s)$ is constructed out of 2^r $B(p,b)$'s. We refer to these $B(p,b)$'s as the components of $B(p+r, b+s)$ and are denoted by $B_i(p,b)$; $1 \leq i \leq 2^r$. Where there is no danger of ambiguity, we refer to $B_i(p,b)$, simply as B_i.

Definitions: Consider $B(p+r, b+s)$ made of 2^r components $B(p,b)$'s.

(i) For any component B_i of $B(p+r, b+s)$, μ_i is the number of changes made to B_i.

(ii) $\Delta(p,b,r,s) = \dfrac{\overset{2^r}{\underset{i=1}{\text{Min}}}\, \mu_i}{2^{p+1}}$

(iii) $\Gamma(p,b,r,s) = \dfrac{\sum_{i=1}^{2^r} \mu_i}{2^{p+r+1}}$

Δ is related to the situation where $B(p,b)$ is the existing configuration, which is being scaled up to $B(p+r, b+s)$ by adding more processors (and perhaps buses). This measure reflects the fraction of connections in $B(p,b)$ (which are 2^{p+1} in number, counting both t and 1 connections) that must be changed to construct $B(p+r, b+s)$. On the other hand,

Γ is related to constructing $B(p+r, b+s)$ out of already existing $B(p,b)$'s. It reflects the fraction of the total number of connections in all the component $B(p,b)$'s (which is 2^{p+r+1} in number) that must be changed to obtain $B(p+r, b+s)$.

For $b+s < 2$, $\Delta(p,b,r,s) = \Gamma(p,b,r,s) = 0$ as all the processors are connected to all the buses. We therefore consider the case where $b \geq 2$. From the constructions 1 and 2, it is clear that $\Delta(p,b,1,1) = \Delta(p,b,1,0) = 0$, for all p, b $(2 \leq b < p)$. From this it follows that

$$\forall r \geq s \geq 0,\ \Delta(p,b,r,s) = 0$$

The remainder of this section is devoted primarily to deriving an expression for Γ. For brevity, we denote $\sum_{i=1}^{r} \mu_i$ in the expression for $\Gamma(p,b,r,s)$ by $\nu(p,b,r,s)$. Clearly, $\nu(p,b,0,0) = 0$. From Fig. 3 we get $\nu(p,b,r,0) \leq 2\nu(p,b,r-1,0)+1$ and from Fig. 2 we have $\nu(p,b,r,r) \leq 2\nu(p,b,r-1,r-1)+1$; for $r > 0$. Therefore

$$\forall r \geq 0,\ \nu(p,b,r,0),\ \nu(p,b,r,r) \leq 2^r - 1 \tag{3}$$

Case 1: $r \geq s \geq 0$.
We first construct $B(p+r-s, b)$ from $B(p,b)$ and then $B(p+r, b+s)$ from $B(p+r-s, b)$. From equation 3 we get

$$\forall r,s;\ r \geq s \geq 0,\ \nu(p,b,r,s) = \nu(p,b,r-s,0) + \nu(p+r-s,b,s,s) \leq 2^{r-s} - 1 + 2^s - 1 \tag{4}$$

Case 2: $s > r \geq 0$. Here we consider two subcases.
Subcase 1: $p > b+s$.
Consider first $B(p,b)$, that is to be scaled up to $B(p, b+s)$. In building $B(p, b+s)$ we first reduce $B(p,b)$ into 2^s component $B(p-s,b)$'s and then use construction 1, s times till we obtain $B(p, b+s)$. In constructing $B(p,b)$ from 2^s components (each a $B(p-s,b)$) the total number of changes is $\nu(p-s,b,s,0) \leq 2^s - 1$ (from equation 3). This is also the number of changes required for reducing $B(p,b)$ into 2^s component $B(p-s,b)$'s. For Δ no other change need be counted. In converting $B(p-s,b)$ to $B(p,b+s)$ the total number of changes is $\nu(p-s,b,s,s) \leq 2^s - 1$ (from equation 3). Finally to construct $B(p+r, b+s)$ from $B(p, b+s)$ the number of changes is $\nu(p, b+s, r, 0) \leq 2^r - 1$ (from equation 3). Thus, we have

$$\forall r,s;\ s > r \geq 0,\ \ni p > b+s,\ \Delta(p,b,r,s) \leq \frac{2^s-1}{2^{p+1}} \leq \frac{1}{2^{p-s+1}} \leq \frac{1}{2^{b+2}} \tag{5}$$

Here, we have used the fact that $p + r > b + s \Longrightarrow p - s + r > b$.

$$\forall r,s;\ s > r \geq 0,\ \ni p > b+s,\ \nu(p,b,r,s) \leq 2(2^s - 1) + 2^r - 1 \tag{6}$$

Subcase 2: $p \leq b+s$.
Here we first construct $B(p, b+s-r)$. Since $p+r > b+s \Longrightarrow p-s+r > b$, subcase 1 can be applied for this construction. Next we construct $B(p+r, b+s)$ from $B(p, b+s-r)$ by r applications of construction 2. Thus, we have

$$\forall r, s;\ s > r \geq 0,\ \ni p \leq b+s,\ \Delta(p,b,r,s) \leq \frac{2^{s-r}-1}{2^{p+1}} \leq \frac{1}{2^{p-s+r+1}} \leq \frac{1}{2^{b+2}} \qquad (7)$$

$$\forall r, s;\ s > r \geq 0,\ \ni p \leq b+s,\ \nu(p,b,r,s) \leq 2(2^{s-r}-1) + 2^0 - 1 + 2^r - 1 \qquad (8)$$

Therefore, $\forall r, s \geq 0,\ \forall p > b \geq 2\ \ni p + r > b + s \geq b$,

if $r \geq s$

$$\Delta(p,b,r,s) = 0$$

$$\Gamma(p,b,r,s) \leq \frac{2^{r-s}-1+2^s-1}{2^{p+r+1}} \leq \frac{2(2^r-1)}{2^{p+r+1}} \leq \frac{1}{2^p} \leq 0.125$$

if $s > r$

$$\Delta(p,b,r,s) \leq \frac{1}{2^{b+2}} \leq 0.0625$$

if $p > b + s$

$$\Gamma(p,b,r,s) \leq \frac{2(2^s-1)+2^r-1}{2^{p+r+1}} \leq \frac{1}{2^{p+r-s}} + \frac{1}{2^{p+1}} \leq \frac{1}{2^{b+1}} + \frac{1}{2^{p+1}} \leq 0.1875$$

if $p \leq b + s$

$$\Gamma(p,b,r,s) \leq \frac{2(2^{s-r}-1)+2^r-1}{2^{p+r+1}} \leq \frac{1}{2^{p+r-s+r}} + \frac{1}{2^{p+1}} \leq \frac{1}{2^{b+r+1}} + \frac{1}{2^{p+1}}$$

$$\leq \frac{1}{2^{b+1}} + \frac{1}{2^{p+1}} \leq 0.1875$$

5. Concluding Remarks

Our network assumes that both the number of processors and buses are integer powers of two. It is possible to extend our method to P processors and B buses (where P and B are not integer powers of 2). Here the time needed to solve the fan-in computation is not optimal but the degradation is within a constant factor of the optimal time.

Our network has two ports per processor and therefore stands the danger of becoming disconnected with two bus failures. An avenue for future research is to design networks that have x connections per processor. This would have more room for better fault tolerance. The scope of the design could also be expanded to include other computations like the *ASCEND* and *DESCEND* classes of algorithms introduced in [9].

References

[1] A. Aggarwal, "Optimal Bounds for Finding Maximum on Array of Processors with k Global Buses", *IEEE Trans. Computers*, Vol. C-35, No. 1, Jan. 1986, pp. 62–64.

[2] D. A. Carlson, "Performing Tree and Prefix Computations on Modified Mesh-Connected Parallel Computers", *Proc. Int. Conf. on Parallel Proc.*, 1985, pp. 715–718.

[3] M. Dubois, "Throughput Analysis of Cache-Based Multiprocessors with Multiple Buses", *IEEE Trans. Computers*, Vol. C-37, No. 1, Jan 1988, pp. 58–70.

[4] M. A. Holliday and M. K. Vernon, "Exact Performance Estimates for Multiprocessor Memory and Bus Interference", *IEEE Trans. Computers*, Vol. C-36, No. 1, Jan 1987, pp. 76–85.

[5] J. Kilian *et al*, "The Organization of Permutation Architectures with Bussed Interconnections (Extended Abstract)", *Proc. 28th Annual IEEE Symp. on the Foundations of Comp. Science*, 1987, pp. 305–315.

[6] M. D. Mickunas, "Using Projective Geometry to Design Bus-Connection Networks", *Proc. ACM/IEEE Workshop on Interconnection Networks for Parallel and Distributed Processing*, 1980, pp. 47–55.

[7] T. N. Mudge *et al*, "Multiple Bus Architectures", *IEEE Computer*, June 1987, pp. 42–48. 1984, pp. 400–408.

[8] D. K. Pradhan, "Fault-Tolerant Multiprocessor Link and Bus-Network Architectures", *IEEE Trans. Computers*, Vol. C-34, No. 1, Jan. 1985, pp. 33-45.

[9] F. P. Preparata and J. Vuillemin, "The Cube Connected Cycles: A Versatile Network for Parallel Computation", Comm. ACM, Vol. 24, No. 5, May 1981, pp. 300–309.

[10] Q. Stout, "Meshes with Multiple Buses", *Proc. 27th Annual IEEE Symp. on Foundations of Computer Science*, 1986, pp. 264–273.

[11] A. Varma, "Combinatorial Design of Bus-based Interconnection Structures", Research Report RC 12550, IBM Research Division, September 1986.

[12] Q. Yang and L. N. Bhuyan, "Analysis of Packet-Switched Multiple-Bus Multiprocessor Systems", *IEEE Trans. Computers*, Vol. C-40, No. 3, March 1991, pp. 352–357.

Shortcut Routing in Deadlock-Free Networks

F.C.M. Lau
Department of Computer Science
University of Hong Kong

S.W. Lau
Computer Science Department
University of California, Los Angeles

1. Introduction

The Acyclic Buffer Graph (ABG) approach (for example, [A,DS]) to deadlock-freedom in packet-switched networks imposes restrictions on routing which limit the choices of paths that can be taken by a packet and may cause uneven traffic patterns [AV], hot spots, and increased packet delays. In this paper, we propose and discuss a general and simple approach to shortcut routing that can reduce the effect of routing restrictions and hence improve the performance of a deadlock-free network based on ABG. We assume in the following a network representable by an undirected graph (all links are bidirectional) with all "direct deadlocks" involving a single link being prevented using a simple atomic exchange protocol.[1] One or more ABG's can be superimposed on this network, thus resulting in a deadlock-free routing network.

2. Shortcut Routing

For simplicity of discussion and without loss of generality, we assume in this paper that there is only one ABG applied to the network, and that the ABG is of the (simplest) kind: an acyclic, connected graph with no structuring of buffers within a node; hence, a packet can flow freely along links that are in the ABG, as opposed to being dictated by direction information associated with different classes of buffers within the nodes in other ABG approaches. It is easy to show that our shortcut approach can be tuned to apply to these latter kinds of ABG.

Definition 1 *A link is green if it is in the ABG; red otherwise. A path (from some current node to the destination) w.r.t. a packet is a green path if all the links are green; red otherwise.*

Note that from every node, at least one of the paths to every other node is a green one.

A packet normally travels along the shortest green path. It could however consider taking a shortcut (a red path) upon arriving at a junction, called a *fork node*, where such a shortcut path begins. Since the shortcut path is red and comprises at least one red link, there exists the risk of not being able to complete the shortcut. Specifically, at the red link (node v), the packet encounters one of several possible scenarios: (1) the node (w) at the other end of the red link has a vacant buffer, (2) w has no vacant buffer but has a packet wanting to cross the red link to travel to v, and (3) w is filled with packets waiting to jump on to their green paths (moving further away from v). Note that there exist no two adjoining red links, because otherwise, the node in the middle would not be reachable in the ABG. For (1), the packet makes a cross right away; for (2) an atomic exchange can be executed to swap the two packets; for (3), one of the following may be tried.

Persistent. The packet insists on crossing the red link; it either waits in v *as long as not all the buffers in* v *are occupied by packets waiting to go to* w, or it may get preempted occasionally by packets destined for v and get "derouted' to some node nearby; in the latter case, it would always try to come back to v.

Non-persistent. The packet retreats and takes a green path to the destination, trying further shortcuts along the way whenever possible.

It is easy to see that neither persistent nor non-persistent packets would induce deadlocks in the network. Note that persistent shortcut routing is similar to adaptive routing where packets insist on waiting for the needed resources, except persistent shortcut routing would not wait on a resource if it would lead to a cyclic waiting. The potential penalty incurred in retreating is linear with the distance traveled before reaching the red link + the length of the green path from the point where the packet cannot go through the red link to the destination − the length of the original green path.

Definition 2 *A path* P_1 *emanating from another path* P_2 *at some fork node is a shortcut path if*

$$D(\textit{fork node, destination, } P_2) > D(\textit{fork node, destination, } P_1)$$

where $D(i,j,P)$ *stands for the distance in hops between node* i *and node* j *along path* P.

---

[1] The atomic exchange algorithm described in [BBG] can be used to exchange the two packets at the ends of the link.

For simplicity, we only consider distance (in hops) as the measure of path length in the above definition; in practice, traffic conditions for example can be taken into account. For the rest of this paper, we will discuss the case of non-persistent shortcut routing which is more appropriate if the shortcut paths are relatively short (hence smaller penalty), and the nodes, especially those at the ends of red links, have a relatively small number of buffers.

The rule for non-persistent shortcut routing can be stated as follows.

Non-persistent shortcut routing. *(A) A packet can enter a shortcut path if the next link along this shortcut path is a green link or, if it is a red link, the node at the end of it can accept this packet (via atomic exchange or has a vacant buffer). (B) A rejected packet must not wait on a red link.*

Definition 6.3 *A path is safe w.r.t. a packet if there exists a green path between each node on the path and the packet's destination; unsafe, otherwise.*

A safe path includes a red path with one red link, but not a path with two red links because a packet that fails to complete an unsafe shortcut path may be trapped between the two red links. Nevertheless, if the ABG is connected (all nodes are reachable from any node), there must exist a green path, albeit a longer one, to the destination from a node in between the two red links, which the trapped packet may take as an escape. In any case though, a packet should only try to take safe shortcut paths.

Next, we should establish a condition to prevent a packet from traveling in a cyclic path indefinitely. A cyclic path emerges if a retreating packet happens to fall back on to a node that precedes the fork node in the original green path. Cyclic paths are harmful if we consider that the shortcut path that a packet just tried and failed is not worth trying any more. Cyclic paths are not possible if the conditions in the following theorem are met .

Theorem 1 *If (1) all the links are bidirectional, (2) distance is measured in hops only, and (3) shortcuts are limited to one level, then cyclic paths due to retreat from shortcut failures would not exist.*

(3) means that once rejected by a red link, a packet should retreat to its original green path without taking any further shortcuts until it reaches the fork node. If we allow multiple levels of shortcuts, that is, shortcut within shortcut, then it is easy to see that cyclic paths might emerge, which is all right if we can make sure that the packet would avoid going through shortcuts that it tried and failed before. This can be done for example by encoding the failed shortcuts using extra bits within the packet.

Theorem 2 *In a network in which multiple levels of shortcuts are allowed, if a packet would avoid going through failed shortcuts again, then the penalty or extra number of hops incurred by shortcut failures is bounded.*

3. Concluding Remarks

Since there may be a penalty of extra hops when a packet fails to go through the red link in a shortcut path, a packet should estimate the chance of success before taking the shortcut path and should not try to take a shortcut path if the "expected" number of extra hops for taking the shortcut path is positive or too great a number. In the special case in which the shortcut path contains exactly one red link, a packet should try to take the shortcut path whenever possible since there is no penalty upon failure.

From the simulation results in [GHKP], if the performance of a network is not degraded, then the buffer utilization is less than a half. Hence, we can conclude that shortcut routing (persistent or non-persistent) is a viable approach since the chance of succeeding a shortcut is high in normal circumstances.

Our approach is general and can be used to derive both Toueg's adaptive routing and Roscoe's shortcut routing methods as described in [T] and [R] respectively.

References

[A] J.K. Annot, "A Deadlock Free and Starvation Free Network of Packet Switching Communication Processors", *Parallel Computing*, Vol. 9, 1988/89. 147-162.

[AV] V.S. Adves and M.K. Vernon. Performance Analysis of Multiprocessor Mesh Interconnection Networks with Wormhole Routing. Tech. Report, CSD, University of Wisconsin, 1991.

[BBG] J. Blacewicz et al., "Deadlock-resistant Flow Control Procedures for Store-and-Forward Networks", *IEEE Trans. on Comm.*, Vol. C-32, No. 8, 1984. 884-887.

[DS] W.J. Dally and C.L. Seitz, "Deadlock-free Message Routing in Multiprocessor Interconnection Networks", *IEEE Trans. on Computers*, Vol. C-36, No. 5, 1987. 547-553.

[GHKP] A. Giessler et al., "Free Buffer Allocation – An Investigation by Simulation", *Computer Networks*, Vol. 2, 1978. 191-208.

[R] A.W. Roscoe, "Routing Messages Through Networks: An Exercise in Deadlock Avoidance", *Proc. 7th Occam User Group Meeting*, 1987.

[T] S. Toueg. The Design of Deadlock- and Livelock-free Packet Switching Networks. PhD Thesis, Princeton University, 1979.

A Generalized Dimension Exchange Scheme for Dynamic Load Balancing in Distributed Memory Multiprocessors

Chengzhong Xu & Francis C.M. Lau
Department of Computer Science
University of Hong Kong

Abstract

In this paper, we present a generalized dimension exchange scheme for dynamic load balancing in distributed memory, point-to-point connected multiprocessors. In this scheme, an exchange parameter λ is introduced to dictate how the workload is distributed between a pair of processors at each time step. We prove that in an arbitrary structure, dynamic load balancing using our scheme will terminate in the balanced state for any initial workload distribution if $\lambda \in (0,1)$. For a given structure, we prove the existence of an optimal λ which would lead to termination at the fastest asymptotic convergence rate, and in particular, $\lambda=1/2$, i.e. the case of equal distribution in the dimension exchange scheme, is the optimal choice for some structures such as hypercube, quadrilateral and 4×4 torus. Finally, we establish a connection between the generalized dimension exchange scheme and the diffusion scheme, and derive a sufficient condition under which the former is more efficient than the latter.

1 Introduction

Dynamic load balancing is a technique essential for improving the performance of parallel processing by efficiently utilizing the processing power of the processors making up the system. It attempts to balance the workload among the processors by transferring some processes at heavily loaded processors to lightly loaded ones during their execution according to the current system state. In the past decade or so, dynamic load balancing in distributed systems has been studied extensively and there have been many proposals [5], most of which, however, have been evaluated using the simulation approach. Recently Cybenko and Hosseini introduced a completely analyzable matrix iterative approach for the design and analysis of two synchronous[1] load balancing policies for distributed memory, point-to-point connected multiprocessors (DMMP for short), the *diffusion scheme* and the *dimension exchange scheme* [3, 6].

In the diffusion scheme, each processor balances its workload with all its directly connected processors simultaneously. In reality, however, the processor can actually communicate with only one neighbor at a time—hence the dimension exchange scheme was introduced. Cybenko proposed it for the hypercube structure, and later on Hosseini *et al* extended it for arbitrary structures. In the dimension exchange scheme, load balancing happens in only one "dimension" at a time (for an arbitrary structure, the dimension is defined by edge-coloring) and results in an *equal* distribution of workload between each pair of processors along the dimension. The overall balancing happens in all dimensions by turns. Using the matrix iterative approach, the workload distribution is treated as a vector, each element of which represents the local workload maintained by a processor; and the changes to be applied to the distribution according to the balancing policy are modeled as a matrix. The whole process of load balancing is simply an iteration through a series of these matrices, and the study of a dynamic load balancing policy is thus reduced to the analysis of its corresponding iterative process.

---

[1] In synchronous systems, all processors are synchronized by a physical or logical global clock, and therefore run in steps [2].

It is possible that with the dimension exchange scheme, equal distribution of workload between a pair of processors at each step might not lead to an optimal convergence rate of the iterative process for a variety of structures such as mesh and ring, and that unequal distribution might result in a better performance. Therefore, we present in this paper a generalized dimension exchange scheme (*GDE* scheme for short) for dynamic load balancing in an arbitrary network. This new scheme is a generalized one because it relies on an *exchange parameter* λ to split the load between processors inste ad of always splitting it in halves.

The rest of the paper is organized as follows. In Section 2, the GDE scheme is presented and modeled as an iterative process defined by a matrix $M(\lambda)$. In Section 3, we analyze the convergence of the iterative process. We prove that the iterative process is convergent for any initial workload distribution if $\lambda \in (0,1)$; and for a given color graph, there exists an optimal $\lambda \in (0,1)$ which would lead to an optimal convergence rate. We also show that for some structures such as hypercube, $\lambda=1/2$, which corresponds to equal distribution, is an optimal choice. In Section 4, we establish a connection between the GDE scheme and the diffusion scheme, and give a sufficient condition under which the former is better than the latter. Finally, some open problems are discussed in Section 5.

2 The Generalized Dimension Exchange Scheme

We first make some assumptions about the underlying computation and the DMMP. The underlying computation is assumed to be composed of a large number of independent processes, which are executed distributively in the DMMP. The basic unit of workload is a process, and one or more processes may be running in a processor at any time. In addition, the total workload in the process of load balancing is assumed fixed, i.e, no processes are created or killed during that period. A DMMP is composed of a finite number of autonomous homogeneous processors connected by a point-to-point communication network. Each processor has a number of bidirectional communication links through which it interacts synchronously with its neighbours. A DMMP is represented by a simple graph G=(V,E), where V is a set of vertices labeled from 1 to n and $E \subseteq V \times V$ is a set of edges. Each vertex represents a processor and each edge $(i,j) \in E$ represents a communication link between processor i and j. For each vertex i in G, the number of links incident on i is called the **degree** of i and denoted by $\delta(i)$. The maximum degree in G is denoted by $\delta(G)$, i.e. $\delta(G) = \max_{i \in V} \delta(i)$. A graph G is said to be **regular** if all of its vertices have the same degree.

Similar to the dimension exchange scheme for an arbitrary network [6], our GDE scheme is based on edge-coloring of G. The edges of G are supposed to be colored beforehand with the least number of colors (k, say), and no two adjoining edges are assigned the same color number. We represent the k-color graph as $G_k = (V_k, E_k)$, where V_k is the same set of vertices as V and E_k is a set of 3-tuples of the form $(i, j; c)$, $(i, j; c) \in E_k$ if c is the color number of the edge $(i, j) \in$ E.

Definition 2.1 *Let G_k be a k-color graph of G. A sequence of edges in G_k of the form*

$$(i = i_0, i_1; c_1), (i_1, i_2; c_2), \ldots, (i_{l-1}, i_l = j; c_l)$$

is called a **color path** *of length l from i to j if all intermediate vertices $i_s (1 \leq s \leq l-1)$ are distinct and $k \geq c_1 > c_2 > \ldots > c_l \geq 1$. A color path from i to j is said to be* **closed** *if i=j. Two color paths fro m i to j in G_k are said to be distinct if their intermediate vertices do not coincide at all. All of the distinct color paths from i to j comprise a set $\mathcal{P}_{ij}$, with a typical element p.*

Fig. 1-4 are 4 examples of 3-color graphs and Fig. 5 is a 4-color graph. Note that the number in bracket is a color number tagged on an edge, and the vertex labels in Fig. 4 and 5 are omitted for simplicity. In the graph of Fig. 3, the sequence of edges (2,3;3), (3,1;2) and (2,1;1) are two distinct color paths from vertex 2 to 1 of length 2 and 1 respectively. The sequence of edges (2,3;3), (3,1;2), (1,2;1) is a closed color path of length 3 incident on vertex i.

Definition 2.2 *Let G_k be a k-color graph of G. G_k is* **k-regular** *if for each vertex i, $1 \leq i \leq n$, the incident edges of i have consecutive color numbers from 1 to k, and $k=\delta(G)$. G_k is* **symmetric** *if for each pair of vertices i, j, $1 \leq i, j \leq n$, there are the same number of distinct color paths between them in each direction, that is, $|\mathcal{P}_{ij}| = |\mathcal{P}_{ji}|$.*

It is easy to show that G_k is k-regular if and only if G is regular and $\delta(G)$=k. Note that the k-regularity of G_k does not immediately follow from the regularity of its underlying graph G. For instance, a ring with odd vertices such as in Fig. 1 is regular of degree 2, but its maximum color number is 3, and hence its color graph is a 3-color graph.

We can now present the GDE scheme based on a k-color graph G_k. We first divide time t into a sequence of time points, t=0,1,2,.... Each processor interacts synchronously with one of its neighbors at each time point. Let w_i be the current local workload of processor i, t be the current time point, and λ be an introduced parameter for a given G_k. Then, the change of local workload in each processor with the GDE scheme is dictated by the following algorithm.

```
Initally, t=0
Procedure for Processor i ($1 \leq i \leq n$)
   while (not terminate) {
         for (tp=t; tp<t+k; tp++) {
               c = (tp mod k)+1;
               for all j, $1 \leq j \leq n$, if (i,j;c) $\in E_k$, then
                     $w_i = (1-\lambda)w_i + \lambda w_j$;
         }
         t=tp;
   }
```

Note that processor i finishes a complete sweep after k consecutive executions of the for-loop, during which it interacts with all of its neighbors one at a time. In order to guarantee $w_i > 0$, the domain of the exchange parameter λ is restricted to [0,1]. As a special case, the GDE scheme coincides with the dimension exchange scheme when λ=1/2.

Two problems are in order. One is the *termination condition* under which the algorithm would terminate at some time. The other is the problem of *efficiency*—that is, how many iterative steps are needed for the algorithm to arrive at its termination state in the worst case. In the following, we analyze the algorithm with respect to these problems.

Iterative Model. Let $w_i^t (1 \le i \le n)$ denote the local workload of processor i at time t. Then, the workload distribution at time t is denoted by a vector $W^t = (w_1^t, w_2^t, \ldots, w_n^t)^T$. W^0 is supposed to be the initial workload distribution. Then, based on the above algorithm, the workload change in processor i at time t and $(t \; mod \; k) + 1 = c$ can be modeled as:

$$\begin{aligned} w_i^{t+1} &= (1-\lambda) w_i^t + \lambda w_j^t \quad if \; \exists_j (i,j;c) \in E_k \\ w_i^{t+1} &= w_i^t \quad otherwise \end{aligned} \tag{1}$$

where $0 \le \lambda \le 1$.

As a whole, the change of the workload distribution of the entire system at time t and $(t \; mod \; k) + 1 = c$ can be modeled as:

$$W^{t+1} = M_c(\lambda) W^t \quad 1 \le c \le k \tag{2}$$

where for $1 \le i,j \le n$ and $i \ne j$,

$$\begin{aligned} &(M_c(\lambda))_{ii} = (M_c(\lambda))_{jj} = (1-\lambda) \; and \; (M_c(\lambda))_{ij} = (M_c(\lambda))_{ji} = \lambda \quad if \; (i,j;c) \in G_k; \\ &(M_c(\lambda))_{ii} = 1 \; and \; (M_c(\lambda))_{ij} = 0 \quad otherwise. \end{aligned}$$

Consequently, the change of the workload distribution of the entire system at time t can be modeled as:

$$W^{t+k}(\lambda) = M(\lambda) W^t \tag{3}$$

which implies

$$W^{tk}(\lambda) = M^t(\lambda) W^0 \quad t = 0, 1, 2, \ldots \tag{4}$$

where $M(\lambda) = M_k(\lambda) \times M_{k-1}(\lambda) \times \ldots \times M_1(\lambda)$, and is called the *GDE matrix* on a k-color graph G_k. Specially, $M(\lambda)$ equals the dimension exchange matrix when λ=1/2.

The above iterative equations (1)-(4) formally model the behavior of dynamic load balancing using the GDE scheme. As the features of the GDE scheme are fully captured by the iterative process determined by $M(\lambda)$, the termination of the algorithm is equal to the convergence of the sequence of workload distribution $\{W^{tk}\}$ generated by equation (4), and the efficiency of the algorithm is reflected by the asymptotic convergence rate.

3 An Analysis of the GDE Scheme

First, we carry out a convergence analysis of the iterative process determined by a GDE matrix $M(\lambda)$. Let $\mu_j(\lambda)(1 \le j \le n)$ be the eigenvalues of $M(\lambda)$, and $\rho(\lambda)$ the spectral radius of $M(\lambda)$. We now define $\gamma(\lambda) = max\{|\mu_j(\lambda)| : 1 \le j \le n, \mu_j(\lambda) \ne 1)\}$. Then, if $\rho(\lambda) < 1$, $\gamma(\lambda) = \rho(\lambda)$;

otherwise, $\gamma(\lambda)$ is the subdorminant eigenvalue of M(λ) in modulus. According to the matrix iterative theory [1], the sequence of $\{M^t(\lambda)\}$ is convergent if and only if $\gamma(\lambda) < 1$. Following this approach, we establish the following convergence theorem.

Theorem 3.1 *Let $M(\lambda)$ be the GDE matrix on a G_k, $\overline{M}$ a matrix of order n in which each element equals 1/n, and $\overline{W}$ a uniform distribution vector in which each element equals 1/n, then*

1. *$\lim_{t\to\infty} M^t(\lambda) = \overline{M}$ iff $\lambda \in (0,1)$.*
2. *For any initial workload distribution W^0, the sequence of W^{tk} generated by our GDE scheme converges to $b\overline{W}$ if $\lambda \in (0,1)$, where $b=\sum_{1\le i\le n}(w_i^0)/n$.*

Proof (Sketch). (1). Firstly, it can be shown that M(λ) is nonnegative and doubly stochastic when $0 \le \lambda \le 1$, and that M(λ) is primitive when $0 < \lambda < 1$. According to the fundamental Perron-Frobenius theorem on nonnegative matrix, we have $\rho(\lambda)$=1 and its algebraic multiplicity is 1 when $0 < \lambda < 1$. Hence, $\gamma(\lambda) < 1$. Thus, $\lim_{t\to\infty} M^t(\lambda)$ exists, and each of its columns is a positive eigenvector of M(λ) corresponding to the spectral radius $\rho(\lambda)$=1. That is, $\lim_{t\to\infty} M^t(\lambda) = \overline{M}$ when $0 < \lambda < 1$. Moreover, because M(0) and M(1) are the identity matrix and a permutation matrix, respectively, both of them have the eigenvalue 1 in modulus with multiplicity n. Hence, $\lim_{t\to\infty} M^t$ does not exist or exists but does not converge to $\overline{M}$. Consequently, we have $\lim_{t\to\infty} M^t = \overline{M}$ if and only if $\lambda \in (0,1)$.
(2). Since $\lim_{t\to\infty} M^t(\lambda) = \overline{M}$, for any initial workload distribution W^0, we have $\lim_{t\to\infty} W^{kt} = \overline{M}W^0 = b\overline{W}$, where $b = \sum_{1\le i\le n} w_i^0/n$. □

The above theorem shows that $0 < \lambda < 1$ is a sufficient and necessary condition for the termination of dynamic load balancing using the GDE scheme. In reality, when λ=0, the workload distribution does not change after an iteration step, and when λ=1, an iteration step results in a complete exchange of the workload between pairs of vertices, and hence, the variance of workload distribution does not change at all.

We now consider the convergence rate of the iterative process defined by M(λ), which is denoted by $R_\infty(M(\lambda))$. Since the spectral radius of M(λ) is unique, the size of the subdorminant eigenvalue $\gamma(\lambda)$ determines the asymptotic convergence rate of $\{M^t(\lambda)\}$ and $R_\infty(M(\lambda)) = -\ln\gamma(\lambda)$. In the following theorem, we prove that given a k-color graph, there exists an optimal λ which leads to the fastest asymptotic convergence rate $R_\infty(M(\lambda))$.

Theorem 3.2 *Let $M(\lambda)$ be the GDE matrix on a G_k. There exists a $\lambda_b \in (0,1)$ such that its asymptotic convergence rate $R_\infty(M(\lambda_b)) \ge R_\infty(M(\lambda))$ for all $\lambda \ne \lambda_b$.*

Proof (Sketch). Since $\gamma(\lambda)$, as an eigenvalue of M(λ), is continuously dependent on the elements of M(λ), $\gamma(\lambda)$ continuously depends on λ because each element of M(λ) is a polynomial in λ when $\lambda \in [0,1]$. Hence, there exists a $\lambda_b \in [0,1]$ such that $\gamma(\lambda_b) \le \gamma(\lambda)$ for all $\lambda \in [0,1]$. Also, because $\gamma(\lambda)$=1 when λ=0 or 1, $\lambda_b \in (0,1)$. □

$R_\infty(M(\lambda))$ depends not only on the structure of M(λ), but also on the size of each positive element. To examine the effects of λ on $R_\infty(M(\lambda))$, we need to work out a computation formula for each element in M(λ) on λ. The proof of the following lemma is based on the concept of color path and the features of each M$_c$(λ).

Lemma 3.1 *Let $M(\lambda)$ be the GDE matrix on a k-color graph G_k. If $0 < \lambda < 1$, then for $1 \leq i,j \leq n$*

$$\begin{array}{ll} m_{ij}(\lambda) = \sum_{p \in \mathcal{P}_{ij}}((1-\lambda)^{r_p}\lambda^{l_p}) & i \neq j \\ m_{ii}(\lambda) = (1-\lambda)^{\delta(i)} + \sum_{p \in \mathcal{P}_{ii}}((1-\lambda)^{r_p}\lambda^{l_p}) & i = j \end{array}$$

where l_p is the length of the color path $p \in \mathcal{P}_{ij}$ which is of the form

$$(i = i_0, i_1; c_1), (i_1, i_2; c_2), \ldots, (i_{l_p - 1}, i_{l_p} = j; c_{l_p})$$

and $r_p = \sum_{s=0}^{l_p}(n_s)$, n_0 being the number of incident edges of i whose color number is larger than c_1; n_{l_p} is the number of incident edges of j whose color number is smaller than c_{l_p}; $n_s(1 \leq s \leq l_p - 1)$ is the number of incident edges of i_s wh ose color number is larger than c_{s+1} and smaller than c_s.

As an example, we list the GDE matrix M(λ) on the color graph as in Fig. 3. $M(\lambda)=$

$$\begin{vmatrix} \lambda^3 + (1-\lambda)^3 & \lambda^2(1-\lambda) + \lambda(1-\lambda)^2 & \lambda^2(1-\lambda) + ld(1-\lambda)^2 & \lambda^2(1-\lambda) + \lambda(1-\lambda)^2 \\ \lambda^2(1-\lambda) + \lambda(1-\lambda)^2 & \lambda^3 + (1-\lambda)^3 & \lambda^2(1-\lambda) + ld(1-\lambda)^2 & \lambda^2(1-\lambda) + \lambda(1-\lambda)^2 \\ \lambda^2(1-\lambda) + \lambda(1-\lambda)^2 & \lambda^2(1-\lambda) + \lambda(1-\lambda)^2 & \lambda^3 + (1-\lambda)^3 & \lambda^2(1-\lambda) + \lambda(1-\lambda)^2 \\ \lambda^2(1-\lambda) + \lambda(1-\lambda)^2 & \lambda^2(1-\lambda) + \lambda(1-\lambda)^2 & \lambda^2(1-\lambda) + \lambda(1-\lambda)^2 & \lambda^3 + (1-\lambda)^3 \end{vmatrix}$$

The result can be verified through direct matrix multiplication.

From the above lemma and by Definition 2.2, we establish a sufficient condition about a k-color graph under which $R_\infty(M(\lambda))$ is larges t when λ=1/2.

Theorem 3.3 *Given a k-color graph G_k, if G_k is k-regular, symmetric and for each pair of vertices i,j, $1 \leq i,j \leq n$ and $i \neq j$, $|\mathcal{P}_{ij}| = |\mathcal{P}_{ii}| + 1$, then λ=1/2 is the best choice.*

Proof (Sketch). Suppose M(λ) be the GDE matrix on G$_k$. If G$_k$ satisfies the conditions mentioned above, then M(λ) is a uniform matrix of which each element equals $\frac{|\mathcal{P}_{ij}|}{2^k}$ when λ=1/2. Hence, $\gamma(\lambda)$=0. Thus, $R_\infty(M(\lambda))$ is largest when λ=1/2. □

It is easy to verify that the structures of quadrilateral, n-cube, and 4*4 torus as illustrated in Fig. 3-5, respectively, are three examples for which λ=1/2 is the optimal choice with the GDE scheme.

4 Comparison with the Diffusion Scheme

In this section, we establish a connection between our GDE scheme and the diffusion scheme of Cybenko [3], and make a comparison between them in the convergence rates of the sequence $\{M^t(\lambda)\}$ and $\{D^t\}$, where D is the diffusion matrix as defined in [3]. Note that M(λ) reflects a complete sweep—i.e. k consecutive iterative steps. Each step involves an I/O communication step for a processor. The diffusion matrix D reflects a single iterative step. However, an iterative step in the diffusion scheme involves $\delta(G)$ I/O communication steps for a processor. Because $\delta(G) \leq k \leq \delta(G) + 1$ [4], a complete sweep of M(λ) and an iterative step of D are comparable. We introduce the concept of an expanded digraph to help establish the connection:

Definition 4.1 *An expanded digraph, denoted $G'_k = (V'_k, E'_k)$, is a directed graph dedu ced from a k-color graph $G_k = (V_k, E_k)$ as follows,*

1. V'_k=V_k

2. $< i, i > \in E'_k$ *for all* $1 \leq i \leq n$

3. $< i, j > \in E'_k$ *if there exists a color path from i to j in* G_k.

As an example, Fig. 6 and Fig. 7 are the the expanded digraphs of Fig. 1 and Fig. 3, respectively. By the definitions of G, G_k and G'_k, we can see that G is an edge-deleted subgraph of G'_k when each undirected edge (i,j) $\in$ E is viewed as a pair of directed edges $< i, j >$ and $< j, i >$.

Based on the expanded digraph G'_k of G, the connection between the GDE scheme and the diffusion scheme can be established. In the next theorem, we show that the GDE matrix $M(\lambda)$ of a G_k is a diffusion matrix of G'_k.

Theorem 4.1 *Let* $M(\lambda)$ *be the GDE matrix on a k-color graph* G_k *and* G'_k *be the expanded digraph of* G_k. *Then,* $M(\lambda)$ *is a diffusion matrix on* G'_k, *that is, for* $1 \leq i, j \leq n$ *and* $i \neq j$, $m_{ij}(\lambda) > 0$ *if and only if there is a directed edge* $< i, j > \in E'_k$, *and* $m_{ii}(\lambda) = 1 - \sum_{1 \leq j \leq n, i \neq j} m_{ij}(\lambda) > 0$.

Proof (Sketch). For $1 \leq i, j \leq n$ and $i \neq j$, $m_{ij}(\lambda)=0$ iff there are no color paths from i to j iff $< i, j > \notin G'_k$. Moreover, there exists a loop for each vertex. Hence, $m_{ii}(\lambda) > 0$ for all $1 \leq i \leq n$. Thus, $M(\lambda)$ is a diffusion matrix on the digraph G'_k. □

From the above theorem, the comparison of the diffusion scheme and the GDE scheme is then reduced to the comparison of the same diffusion scheme but on two different graphs G and G'_k. Since G is an edge-deleted subgraph of G'_k when G is viewed as a digraph, the diffusion matrix of G'_k has more positive elements than the diffusion matrix of G. That is, a processor can balance its workload only with its neighbors at an iterative step in the diffusion scheme; in contrast, it might balance its workload with more than its neighbors at each complete sweep in the GDE scheme.

Finally, we show that the optimal GDE scheme is more efficient than a diffusion scheme for a color graph G_k satisfying certain conditions. The proof directly follows from the proof of Theorem 3.3.

Theorem 4.2 *Let* $M(\lambda)$ *be the GDE matrix on a k-color graph* G_k *of G, and D be the diffusion matrix of G. Under the conditions of Theorem 3.3, the convergence rate of* $M(\frac{1}{2})$ *is equal to or larger than that of D.*

5 Concluding Remarks

In this paper, we presented a generalized dimension exchange scheme for the dynamic load balancing in distributed memory multiprocessors. We studied analytically the behavior of dynamic load balancing using our scheme by modeling it as an iterative process. Similar to the open problem in the diffusion scheme of determining an optimal diffusion parameter α_{ij}, the choice of an optimal exchange parameter λ for an arbitrary structure using the GDE scheme has yet to be determined. Furthermore, *asynchronous* load balancing should also be thoroughly analyzed,[2] and compared with its synchronous counterpart in terms of the execution and communication complexities.

---

[2] Bertsekas and Tsitsiklis did discuss the convergence of asynchronous load balancing but not its convergence rate [2].

References

[1] Berman, A., and Plemmons, R.J., *Nonnegative Matrices in the Mathematical Sciences*, Academic Press, 1979.

[2] Bertsekas, D.P., and Tsitsiklis, J.N., *Parallel and Distributed Computation*, Prentice-Hall Inc. 1989

[3] Cybenko, G., "Dynamic Load Balancing for Distributed Memory Multiprocessors", *J. of Parallel and Distributed Computing* 7, 279-301(1989).

[4] Fiorini, S., and Wilson, R.J., "Edge-Coloring of Graphs", in *Selected Topics in Graph Theory*, Ed. Beineke, L.W. and Wilson, R.J., Academic Press, 1978.

[5] Harget, A.J., and Johnson, I.D., "Load Balancing Algorithms in Loosely-Coupled Distributed Systems: A Survey", in *Distributed Computer Systems*, Ed. by Zedan, H.S.M., Butterworths, 1990.

[6] Hosseini, S.H., *et al*, "Analysis of a Graph Coloring Based Distributed Load balancing Algorithm", *J.of Parallel and distributed Computing 10*, 160-166(1990).

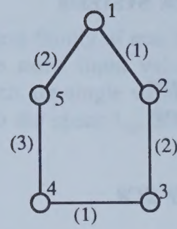

1
(2)
(1)
5
2
(3)
(2)
4
3
(1)

Fig.1 5-Ring

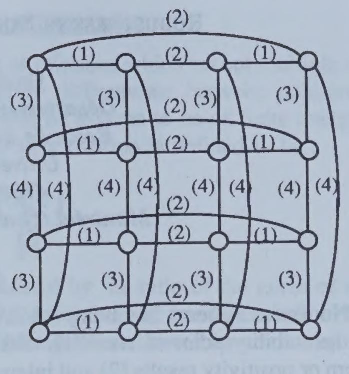

(2)
(1)
(2)
(1)
(3)
(3)
(2)
(3)
(3)
(1)
(2)
(1)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(4)
(2)
(1)
(2)
(1)
(3)
(3)
(2)
(3)
(3)
(1)
(2)
(1)

Fig.5 4*4 Torus

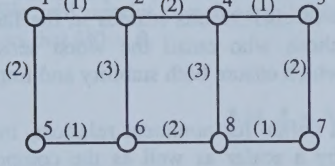

1
(1)
2
(2)
4
(1)
3
(2)
(3)
(3)
(2)
5
(1)
6
(2)
8
(1)
7

Fig.2 2*4 Mesh

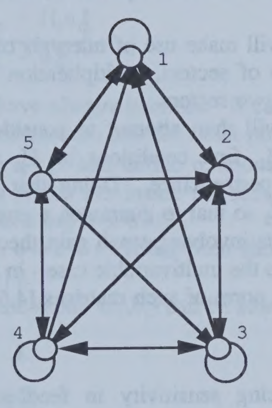


Fig.6 Expanded 5-Ring

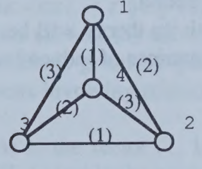

1
(1)
(3)
4
(2)
(2)
(3)
3
(1)
2

Fig.3 Quadrilateral

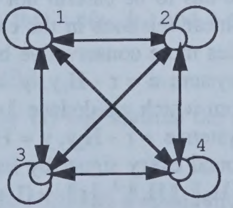


Fig.7 Expanded Quadrilateral

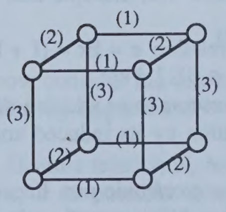

(1)
(2)
(2)
(1)
(3)
(3)
(3)
(3)
(1)
(2)
(2)
(1)

Fig.4 3-Cube

ROBUSTNESS IN NONLINEAR FEEDBACK SYSTEMS

David Bensoussan
Département de Génie électrique
École de technologie supérieure
Université du Québec
4750, Henri-Julien
Montréal (Québec) Canada H2T 2C8

ABSTRACT

Nonlinear systems involving conic/sector non linearities have been studied extensively from the stability point of view [1]. Stability criteria typically make use of the small gain theorem or positivity results [2] and interesting extensions to multivariable systems have been proposed by Safonov [3]. One principal drawback of such results resides in the fact that ultimate boundary stability results are precisely those who entail the worst sensitivity performance. It is our objective to develop criteria which ensure both stability and improved sensitivity.

We will make use of intervals of a sector and define for nonlinear relations, induced norms sum of sectors, multiplication of a sector by a scalar as well as the composition product of two sectors.

We will then attempt to consider the following problem: given a plant H_1 and a feedback H_2, find conditions on H_2 which guarantee, a) stability, b) both stability and sensitivity performance. Doing that we will focus on the penalty on the sector of the feedback H_2 so that to guarantee a given sensitivity upper bound.

Results involving small gain theorem as well as positivity theory will be derived. An extension to the multivariable case - in which we consider matrices of induced norms as well as a matrix norm of such matrices [4,5] - will follow.

I. INTRODUCTION

Reducing sensitivity in feedback systems implies some upper bound on some endogenous signals. In the case of nonlinear systems, those bounds can be better assessed in the case of normable non linearities. The definition of induced norms on conic/sector non linearities [1] makes the choice of such non linearities ideal for the finding of upper bounds. However, one has to be careful not to use the right distributivity property which does not hold for nonlinear relations in the case of a nonlinear system with multiple non linearities, and this implies more conservative bounds.

In the system $e = r - H_2 y$, $y = H_1 e$, the sensitivity relation $e = Er = (I + H_2 H_1)^{-1} r$ is immediate from which we deduce $\|e\| \leq \|(I + H_2 H_1)^{-1} r\| \leq \|E\| . \|r\|$.

In the system $e = r - H_2 y$, $y = H_1 e + d$ in which d represents some additive disturbance, one has to bound every signal by its norm and every relation by its induced norm so that $\|e\| \leq (I + \|H_2\| . \|H_1\|^{-1} \|r\| + (I + \|H_2\| \ \|H_1\|)^{-1} \|d\|$.

Most stability criteria for [1,2,3] systems involving sector/conic non linearities have been preoccupied by bounded input - bounds output (BIBO) stability as well as the continuity of the outputs. But very little has been done in terms of reducing sensitivity. In fact, most BIBO stability results are formulated in such a way that they entice unaffordable sensitivity bounds. It is our objective to review these stability criteria in the light of some sensitivity bound considerations.

II. MATHEMATICAL PRELIMINARIES

The largest family of non linearities is the one of relations which accept multiple output values for the same input values. We will therefore differentiate between relations and operators which are single valued. All our inputs will be assume to be of finite energy, i.e. they belong to the space L_2, and the induced norms of a relation is defined to be :

$$\| H_1 \| - \sup_{\substack{u \in L_2 \\ u \neq 0}} \frac{\| H_1 u \|}{\|u\|}$$

Similarly, the incremental induced norm is defined by the ratio of the norm of output differences over the norm of the corresponding input differences. It has been shown [1] that results involving incremental induced norm imply boundedness as well as continuity of the output.

We will restrict ourselves to relations H_1 having a zero element in both domain and range so that $H0 = 0$.

$$\| H_1 \| - \sup_{\substack{u_1, u_2 \in L_2 \\ u_1 \neq u_2}} \frac{\|H_1 u_2 - H_1 u_1\|}{\|u_1 - u_2\|}$$

Relations do not constitute a linear space and have always inverses. It can be easily shown that they verify the product inequality. We will assume that the inner product space $<x_1, x> = \int_0^\infty x(t)\, y(t)\, dt$ induces the norm $<x_1, x> = \| x \|^2$ and define relations which are interior conic with center c and radius r or relations which are inside the sector (a, b) as follows : $\| Hx - cx \| < r\ \| x \|$ (1) or $<(Hx - ax), (Hx - bx) > \leq 0$ (2).

Equations (1) and (2) are equivalent if $a = c - r$, $b = c + r$. It has been shown [1] that the induced norm is given by $\| H \| \leq \max (| a |, | b |)$.

Operations involving relations H_1 belonging to the sector (a_1, b_1) can be summarized as follows :

1) I is in the sector (1, 1);
2) kH_1 is inside the sector (ka_1, kb_1);
3) $H_1 + H_2$ is inside the sector $(a_1 + a_2, b_1 + b_2)$
4) H^{-1} is outside the sector $(1/b, 1/a)$ if $a > 0$;
   H^{-1} is inside the sector $(1/b, 1/a)$ if $a < 0$.
5) the product sector $(a_1 b_1) \times (a_2, b_2)$ of two relations is defined by the sector $(a_1 b)$ which includes all the real numbers xy such that $x\ \varepsilon(a_1, b_1)$ and $y\ \varepsilon(a_2, b_2)$.

III. SENSITIVITY RESULT

We now consider the feedback system depicted by (1) and address the problem of a desired sensitivity upperbound.

Theorem : Given a relation H_1 belonging to the sector (a_1, b_1), and a sensitivity upperbound ε, it is possible to choose a relation H_2 outside a sector (a_2, b_2) such that $\| (I + H_2 H_1)^{-1} \| < \varepsilon$ provided that : $a_2 = (-1 - 1/\varepsilon)/a_1$, $b_2 = (-1 + 1/\varepsilon)/b_1$ if $a_1 > 0$ and $a_2 = (-1 - 1/\varepsilon)/b_1$, $b_2 = (-1 + 1/\varepsilon)a_1$ if $a_1 < 0$.

Proof : Let H_1 and H_2 be sector relations. Our objective is to get a sensitivity upperbound ε such that $\| E \| \leq \varepsilon$ i.e we are interested that the relation $E = (I + H_2 H_1)^{-1}$ to be in the sector $(-\varepsilon, \varepsilon)$. It follows that $(I + H_2 H_1)$ is outside the sector $(-1/\varepsilon, 1/\varepsilon)$, and that $H_2 H_1$ is outside the sector $H_2 H_1$ is outside the sector $(-1 - 1/\varepsilon, -1 + 1/\varepsilon)$.

Assuming H_1 belongs to the sector (a_1, b_1), $a_1 > 0$, our aim is to find the sector (a_2, b_2) which will ensure that H_2H_1 is outside the sector $(-1 - 1/\varepsilon, -1 + 1/\varepsilon)$. H_2 must be outside a sector (a_2, b_2), $a_2 < 0$, $b_2 < 0$ in order to get a product sector with negative bounds and such that $b_1b_2 = -1 + 1/\varepsilon$ and $a_1a_2 = -1 - 1/\varepsilon$ from which we deduce :

$$b_2 = \frac{-1 + \frac{1}{\varepsilon}}{b_1}, \quad a_2 = \frac{-1 - \frac{1}{\varepsilon}}{a_1}$$

Remark 1

In the case of multiple inputs, it is possible to derive directly the sector bound on H_2 :

$$(I \| H_2 \| \cdot \| H_1 \|)^{-1} < \varepsilon \qquad \| H_2 \| > \frac{-1+\frac{1}{\varepsilon}}{\| H_1 \|} = \frac{-1 + \frac{1}{\varepsilon}}{\max (| a_1 | , \ | b_1 |)} = \Delta$$

i.e. H_2 should be limited to the sector $(-\Delta, \Delta)$.

Remark 2

In a typical BIBO stability result, ε can be any finite number. Very high values of ε would indeed lead to a wider sector $(a_2, b_2) = (-1/b_1, -1/a_1)$ ensuring stability but definitely no sensitivity upper bound.

Remark 3

The present result focusses on reducing some uppersensitivity bound ε and could in theory be applied to $\varepsilon < 1$ if we assume that at high frequencies the sector condition holds. If it is not the case, the sensitivity upper bound ε must be bigger than 1. In cases in which the non linearity is very well defined at high frequency and the higher harmonics and intermodulation frequencies are well prescribed, the result could be applied for $\varepsilon < 1$ under some frequency domain of interest.

Remark 4

A straightforward extension to the multivariable case can be gotten by extrapolating existing results on multivariable systems including non linearities [4,5].

IV. REFERENCES

[1] G. Zames, "*On the Input/Output Stability of the Varying Nonlinear Feedback Systems*", IEEE Transaction on Automatic Control, Vol. ACII, Part I, April 1966, pp. 228-238, Part II, July 1966, pp. 466-476.

[2] A. Michel and R. Miller, "*Qualitative Analysis of Large Scale Dynamical Systems*", Mathematics in Science and Engineering, Vol. 134, Academic Press.

[3] M.G. Safonov, "*Stability and Robustness of Multivariable Feedback Systems*", MIT Press.

[4] D. Bensoussan, "*Robustesse et Commande Décentralisée de Systèmes Multivariables Non Linéaires*", Proceedings of the 7th International Conference on Analysis and Optimization of Systems, Springer Verlag, Vol. 83, June 25-27, 1986, pp. 630-645.

[5] D. Bensoussan, "*Decentralized Control in Nonlinear Systems*", Proceedings of the 25th IEEE Conference on Decision and Control, Athens, Greece, Dec. 10-12, 1986, pp. 865-866.

Biped Control Via Nonlinear Dynamics

Hatem M. Hmam Dale A. Lawrence

Department of Electrical and Computer Engineering
University of Cincinnati
Cincinnati, Ohio 45221-0300

October 1, 1991

Abstract

This paper addresses the gait and stability analysis of a biped vehicle. Though the mechanical model is simple, it captures the essential dynamics of realistic legged locomotion systems. Similar to biological systems, the legs in this system are assumed to have viscoelastic components and are actuated by control forces. The desired gait of the biped system is obtained in a "natural" way, by matching the mechanical properties of the biped with simple nonlinear control laws. The harmonic limit cycle behavior of the controlled system provides smooth locomotion gaits, and generation of different gaits, such as walking and trotting, is achieved by adjusting only three control parameters. Also, the control laws act in concert with the natural oscillatory motion of the mechanical system to provide robust gaits in the presence of disturbances. Hence, large control efforts are not required, and the additional energy needed to obtain locomotion is significantly reduced. Such a control approach can substantially increase the operating time and efficiency of locomotion vehicles which depend on on-board energy supplies.

1 Introduction

Recently a great deal of research has been focused on understanding the principles governing legged locomotion. This research covers a wide range, from the lower levels of biomechanics [1,2,3,4], to the higher levels of neural organization and intelligence [5][6]. The driving motivation for this research extends beyond the desire to understand locomotion in nature. Artificial legged vehicles have important mobility advantages over wheeled vehicles in negotiating rough terrain or cluttered environments [7]. Also, legged locomotion research may provide important data for applications in functional electric stimulation (FES), resulting in hybrid natural/artificial locomotion systems which allow paraplegic patients to regain some of their mobility.

Many control strategies for walking vehicles have been reported [7]. In *finite state control*, the motion cycle is partitioned into a finite number of adjacent equilibrium states with overlapping regions of attraction [8]. The locomotion cycle is simply a transition from one stable state to another. In *Model reference control*, all joint angle trajectories are specified beforehand, and the problem is reformulated to solve for the joint torques which achieve the desired motion.

Unlike the previously mentioned control strategies, the strategy of this paper is to construct feedback control laws using the properties of limit cycles to generate robust, "natural" locomotion. This method also has the advantage that the details of locomotion control are isolated at a "biomechanical" level. The overall behavior, e.g. step height, gait speed, gait type, etc., are controlled by manipulating a few key control parameters. This facilitates the incorporation of "intelligent" capabilities, which need only manipulate control parameters as often as the overall vehicle objectives change. On similar work, Bay and Hemami [9] have used coupled Van Der Pol oscillators in generating rhythmic patterns similar to human gaits, and verified the generation of walking and jumping type motion in simulation studies. However, the dynamic system was not derived from modeling realistic legged vehicles, and an analysis of the stability and robustness of the method was not provided. Katoh and Mori [10] used the limit cycle concept in generating a walking cycle and physically tested the algorithm on the biped machine BIPMAN. It is not clear how their performance analysis applies in practice, since their "morphological" assumptions are not discussed, nor is it shown if the BIPMAN dynamics satisfy these assumptions.

In this paper, we consider a biped system moving vertically in the frontal plane. Even though this is a relatively simple model of a locomotion system, it captures the essential features of legged locomotion. We introduce a specific type of nonlinear control law which provides stable, smooth harmonic motion which is also robust in the presence of environmental disturbances. Section 2 provides a list of the motion equations in different phases. Section 3 introduces the nonlinear control techniques used in controlling the motion of the biped system. The next three sections discuss the control strategy for each phase. Section 7 illustrates a rigorous nonlinear analysis of the gait stability in the phase plane. Finally, some concluding remarks and suggested future work are listed in the conclusion section.

2 Equations of Motion

A closer look at a biped locomotion reveals that three different gait phases may occur in a walking or running cycle. These are the double stance, the single stance, and the ballistic phases. The system considered is constrained to move vertically in the frontal plane (see Figure 1). The system is characterized by a "body" sitting on two massless legs, with intrinsic damping c and stiffness k.

As a convention in subsequent sections, we will make use of indices $i, j \in \{1, 2\}$ with $i \neq j$ where i indexes one leg and j indexes the other leg. Also, let $l_c = l_{nominal} - \frac{Mg}{2k}$ be the leg length at equilibrium when the body is resting on the two legs. The dynamics governing the bipedal motion (see Figure 1) in each phase can be derived using either the Lagrangian or Newton laws. The equation of motion in the double stance phase is:

$$M\ddot{y} + 2c\dot{y} + 2ky = f_{c_i} + f_{c_j} \tag{1}$$

where y is the deviation from resting equilibrium and f_{c_i}, f_{c_j} are the leg control forces. The impact forces on the ground are

$$\begin{aligned} f_i &= c\dot{y} + ky + \tfrac{Mg}{2} - f_{c_i} \\ f_j &= c\dot{y} + ky + \tfrac{Mg}{2} - f_{c_j} \end{aligned} \tag{2}$$

Summing these three equations gives $M\ddot{y} = Mg - (f_i + f_j)$. For the single stance phase, the equation of motion of the stance leg becomes

$$M\ddot{y} + c\dot{y} + ky = f_{c_j} + \frac{Mg}{2} \tag{3}$$

where f_{c_j} is the active force of the stance leg. Notice that the quantity y is measured with respect to the resting equilibrium of the double stance phase. For this reason, the amount

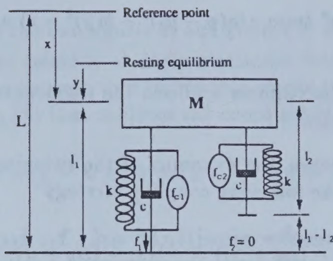


Figure 1: Dynamical model of the biped vehicle

of $Mg/2$ is added to the motion equation of the single stance phase to compensate for the shift of equilibrium positions. The stance leg contact force with the ground is

$$f_j = c\dot{y} + ky + \frac{Mg}{2} - f_{c_j} \tag{4}$$

The swinging leg dynamics are described by

$$c\dot{x}_i + kx_i + \frac{Mg}{2} - f_{c_i} = 0 \tag{5}$$

where $x_i = (l_c - l_i)$ and l_i is the length of the swinging leg. The clearance of the swinging leg from the ground is $l_j - l_i$.

Finally, the ballistic phase is described by the following equation of motion

$$\ddot{x} = -g \tag{6}$$

where x is the position of the body from a given fixed reference. The variable x (see Figure 1) is used instead of y because the dynamics of the body are no longer affected by the dynamics of the legs.

The swinging legs move subject to the dynamical equations

$$c\dot{x}_i + kx_i + \frac{Mg}{2} - f_{c_i} = 0 \tag{7}$$

3 Control Strategy

The control method adopted here relies heavily on the properties of continuous dynamical systems [11][12]. Among other things, we focus on differential equations of the following type to generate a stable periodic motion.

$$M\ddot{y} + \epsilon\left(a(y - y_0)^2 + b(\dot{y})^2 - r\right)\dot{y} + \kappa(y - y_0) = 0 \tag{8}$$

This differential equation has solutions with the following properties.

- If we let $\omega = \sqrt{\frac{\kappa}{M}}$, and choose the coefficients a, b such that $a = b\omega^2$, then periodic solutions are of the form

  $$y = y_0 + \sqrt{\frac{r}{a}}sin(\omega t + \varphi)$$

  where φ is arbitrary.

- The nonlinear "damping" term $\epsilon\left(a(y-y_0)^2+b(\dot{y})^2-r\right)$ approaches zero asymptotically.
- Changing the value of the stiffness κ affects the frequency of leg stepping or equivalently the speed of the gait.

During the single stance phase, the dynamics of the swinging leg are coordinated with those of the stance leg using the following coupling strategy

$$\overbrace{(x_i-(\alpha y+\beta))}^{\cdot}=-\lambda(x_i-(\alpha y+\beta)) \tag{9}$$

where α corresponds to the ratio amplitudes of $\frac{x_i}{y}$, β is an offset between the two signals, and λ determines the rate of convergence of x_i to the quantity $\alpha y+\beta$. The above nonlinear dynamical objectives and coordination control law will be used throughout the following sections to obtain locomotion control for the biped vehicle.

4 Control method of the double stance phase

The control method in this phase is focused on how the two stance legs communicate between each other to achieve a smooth body motion as well as alternating leg strides for subsequent phases of a given gait. The position and velocity of the body motion are sensed and used in a feedback law of the type:

$$f_{c_1}=c\dot{y}+\frac{1}{2}ky+\epsilon\left(r_1-a_1(y-y_0)^2-b_1(\dot{y})^2\right)\dot{y}+\frac{Mg}{4} \tag{10}$$

$$f_{c_2}=c\dot{y}+\frac{1}{2}ky+\epsilon\left(r_2-a_2(y-y_0)^2-b_2(\dot{y})^2\right)\dot{y}+\frac{Mg}{4} \tag{11}$$

where $y_0=\frac{Mg}{2k}$ corresponds to the unstable equilibrium state. The control force parameters a_i, b_i are chosen so that $(a_1+a_2)=(b_1+b_2)\omega^2$. Substituting (10) and (11) into (1), taking $\kappa=k$ yields (8), hence $\omega=\sqrt{\frac{k}{M}}$. The period of the limit cycle is $T=\frac{2\pi}{\omega}$ and the oscillation amplitude is $\sqrt{\frac{r_1+r_2}{a_1+a_2}}$. A natural way of controlling the convergence rate to the stable periodic motion is through the parameter ϵ.

5 Control method of the single stance phase

The dynamics of the stance leg are controlled to have the same periodic motion as in the case of double stance phase. This has the advantage of preserving a smooth ride of the body on the legs as the system change stance phases. The single stance phase has been divided into a passive mode and an active one. During the passive mode, no control force is applied and the landing leg behaves purely as a viscoelastic device. Subsequently the active mode employs a control force in order to sustain the desired limit cycle. The transition between these two modes is triggered at the onset of a slightly positive value of the active control force defined by

$$f_{c_j}=c\dot{y}+\epsilon\left(r-a(y-y_0)^2-b(\dot{y})^2\right)\dot{y} \tag{12}$$

where $r=r_1+r_2$, $a=a_1+a_2$, and $b=b_1+b_2$ The motion of the stance leg is given by substituting (12) into (3). This again results in (8) and the motion asymptotically approaches the stable limit cycle.

In the mean time, the swinging leg should move in phase with the stance leg in order to meet the ground in synchrony just before the onset of a new stride. Following (9), if

$\beta = (1-\alpha)(y_0 - \sqrt{\frac{r}{a}})$, then the two signals x_i and y meet at their minimum, or equivalently the lengths of the two legs coincide at their maximum lengths. At that moment, both legs are in contact with the ground, and the double stance phase begins. The active force applied to the swinging leg (5) that achieves the coordinated motion (9) is

$$f_{c_i} = (k - \lambda c)x_i + \alpha c(\dot{y} + \lambda y) + c\lambda\beta + \frac{Mg}{2} \tag{13}$$

6 Control method of the Ballistic phase

In this phase, the particular flight dynamics for the legs are those which allow for leg stride alternation. Hence, the following control strategy is adopted:

- At the onset of flight phase, stretch the swinging leg using the active force

  $$f_{c_i} = k'(l_{nominal} - l_i) \tag{14}$$

  where $l_{nominal}, l_i$ are the nominal and the actual length of the first leg. k' is a control parameter chosen relatively smaller than k to control the stretching rate of l_i to $l_{nominal}$.

- Contract the most recent stance leg according to

  $$f_{c_j} = k(l_{nominal} - l_j) + \delta(l_j - l_{min}) \tag{15}$$

  where l_j is the actual length of the second leg, δ is a small positive number that controls the contraction rate of the leg. The amount l_{min} is the allowed minimum leg length.

7 Stability analysis of the reduced system

In this section, only the stability analysis for the trotting motion is provided. An analogous stability analysis can be carried out for the walking process. The different dynamical phases invoked during trotting are the single stance and ballistic phases. Before stating the theorem concerning the trotting motion, it is instructive to provide an intuitive exposition of the problem. Since the trotting motion involves large migration of the body from the equilibrium position, the control parameter $r = r_1 + r_2$ is chosen large so that the biped system is forced to enter the ballistic phase. The damping coefficient c of the dissipative process is assumed small to avoid system overdamping. The thick solid line in the lower half of the phase plane corresponds to the body take off ($f_j = 0$). Moreover, the convergence rate λ in (9) is chosen large so that after a small finite time $x_i \approx \alpha y + \beta$. Hence, when the ballistic phase occurs, the deformation of the swinging leg could be approximated by $x_i = \alpha y + \beta$. Finally, since the leg alternation dynamics in the ballistic phase (see section 6) are a deterministic process, then the system state when the biped vehicle lands is well defined. Experimentally, for a given setup of the control parameters, we compute the location of the landing line in the phase plane (see Figure 2).

It will be assumed that the biped system operates safely if

- The starting phase is the single or the double stance phase, and the initial state is inside the region delimited by $(ABCDEFA)$ in Figure 2, or
- The starting phase is the ballistic phase, and the first "landing system state" occurs in the region $(ABCDEFA)$.

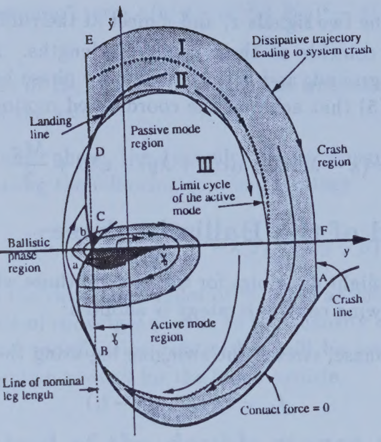

ẏ
E
I
Dissipative trajectory leading to system crash
II
Landing line
Passive mode region
D
Crash region
III
Limit cycle of the active mode
C
b
Ballistic phase region
a
y_0
F
y
A
Crash line
Active mode region
y_0
B
Line of nominal leg length
Contact force = 0

Figure 2: Phase portrait of the biped dynamics

The region $(ABCDEFA)$ will be called the single stance region in the following discussion.

Before stating the theorem, we introduce a useful claim needed in the proof of the theorem.

Claim 7.1 *The ballistic phase begins on the ascending portion of the curve $f_j = 0$.*

Proof: Any point on the curve $f_j = 0$ satisfies

$$ky + \frac{Mg}{2} + \epsilon\left(a(y - y_0)^2 + b(\dot{y})^2 - r\right)\dot{y} = 0$$

But then from (3) and (12), the motion equation is

$$M\ddot{y} + \epsilon\left(a(y - y_0)^2 + b(\dot{y})^2 - r\right)\dot{y} + ky = \frac{Mg}{2}$$

Hence, after substitution of the "damping" term, we obtain $\ddot{y} = g$, that is $\dot{y}$ is increasing. □

Proposition 7.1 *If the biped initial state $(y, \dot{y})$ satisfies*

- *The starting phase is the single stance phase and the initial system state is inside the single stance region.*
- *The initial state is different from $(y_0, 0)$*

then the only possible gait is the trotting motion.

Proof: The point $(y_0, 0)$ is an unstable node for the active dynamics and is a stable focus for the dissipative process. The active mode process has also a stable limit cycle as described in previous sections. The arc (ab) is the ballistic trajectory corresponding to the simultaneous landing of both legs. If the initial state is started in the dark gray area of the phase plane in Figure 2, then after the ballistic phase is completed, the biped system lands with the same stance leg that initiated the take off. But since the "landing state" occurs in the light shaded area, the system state propagates as shown in Figure 2 and eventually exits the light gray region, starts a ballistic phase, and lands (with the other swinging leg) on the "landing line" in region **(III)**. Hence, if the biped system state is initially in the central shaded area, then it propagates "outwards" and eventually flows into region **(III)**.

On the other hand, if the initial state is in the outside shaded region say in the subregion **(I)**, the system state evolves as shown in the Figure 2 and lands after a finite time in the subregion **(II)** or **(III)** because of the claim. But then, any state flow from region **(II)** ends up in the subregion **(III)**. Hence, the system state propagates "inwards" from the outside shaded region into region **(III)**. Thus, there must be at least one stable limit cycle in region **(III)**, and it follows that a portion of the limit cycle corresponds to the ballistic phase (with leg landing alternation) and the rest is associated with the single stance phase. Hence, the only possible gait is the stable trotting gait. □

Proposition 7.2 *If the initial system state satisfies*

- *The starting phase is the double stance phase and the initial state is in the single stance region*
- *The initial state is different from* $(y_0, 0)$

then, the only possible gait is the trotting motion.

Proof: Since the biped system is standing on both legs then both contact forces are positive, that is

$$f_1 = \frac{1}{2}ky + \frac{Mg}{4} - \epsilon\left(r_1 - a_1(y-y_0)^2 - b_1(\dot{y})^2\right)\dot{y} \geq 0$$

$$f_2 = \frac{1}{2}ky + \frac{Mg}{4} - \epsilon\left(r_2 - a_2(y-y_0)^2 - b_2(\dot{y})^2\right)\dot{y} \geq 0$$

Hence,

$$f_1 + f_2 = ky + \frac{Mg}{2} - \epsilon\left(r - a(y-y_0)^2 - b(\dot{y})^2\right)\dot{y} \geq 0$$

But, this expression is exactly that of the reaction force of the single stance phase. Thus, the region where the double stance phase occurs is inside the single stance phase domain (active mode). Since, initially the system state is in the single stance region, and since the dynamics of the body in the double stance phase is identical to that in the case of the single stance phase, then necessarily the biped system switches to the single stance phase while its state $(y, \dot{y})$ is inside the single stance region. Hence, by the last proposition, the only possible subsequent motion of the biped system is the stable trotting gait. □

Theorem 7.1 *If the initial system state* $(y, \dot{y})$ *satisfies*

- *The initial system state is inside the safety region.*
- *The initial state is different from* $(y_0, 0)$

then, the only possible gait for the biped system is trotting.

Proof: The proofs for the trotting gait in the case of the double and single stance phase were given in the last two propositions. In the case of ballistic phase, the "landing system state" by hypothesis is inside the single stance region , therefore by the last two propositions, the subsequent motion is a stable trotting gait. □

8 simulation

The computer simulations of the performance of the actual biped system has been written in MATLAB. The second and third order Runge-Kutta method was used to solve the nonlinear motion equations. The actual simulation was performed on a legged system with $g = 10N/Kg$, $l_{nominal} = 0.5m$, $M = 2Kg$, $L = 1m$, $k = 100N/m$, $c = 4Ns/m$, $a_1 = 200$, $b_1 = 0$, $a_2 = 1800$, $b_2 = 40$, and $\alpha = 1.2$.

The performance of the simulated system has been tested on two different gaits namely walking and trotting vertically in the frontal plane. As shown in Figure 3, There exist a stable limit cycle for the trotting motion ($r_1 = 20$, $r_2 = 180$). The parameter ϵ when chosen large, speeds up the onset of the steady periodic motion, but loads the system with excessive large control forces. A smaller value of ϵ causes the transient motion to survive longer, but has the advantage of alleviating the system from experiencing large force magnitudes. An optimal value of $\epsilon = 0.2$ was chosen. Figure 3 depicts the harmonic behavior of the body motion. Since the biped system is intended to react subject to its intrinsic dynamics, the control effort needed for the stance leg to sustain the periodic motion is small. But, because a large control effort is needed to compress the spring of the swinging leg to clear the ground, the actual model of the swinging leg is not energetically efficient. However, in this paper, the use of nonlinear techniques and their properties were emphasized rather than the discussion of the selective process of the most efficient biped model. Figure 3 shows the time history of the reaction and control forces.

To demonstrate the robustness of the biped motion, the trotting gait is tested on a randomly varied ground level. Figure 4 illustrates the adaptation of the biped system to a challenging rough environment. As seen in the figure, the body enjoys a smooth ride on the legs. This feature is made possible because of the "intelligent" nonlinear feedback laws which quickly adapts the system to the new environment. The peak magnitude of the ground reaction at each leg impact is about three times the weight of the body.

For the walking gait, the control parameters are selected as $r_1 = 1$, $r_2 = 4$, $\epsilon = 40$, and the remaining parameters are kept the same. The body central mass experiences small vertical migrations (see Figure 5), therefore a small control effort is needed to sustain the motion. Moreover, the leg clearance is very small as in the case of normal human walking. The negative peak $\approx -12N$ of the control forces corresponds to the active mode dynamics of the single stance leg. The positive peak corresponds to the control effort needed to compress the spring (k=100N/m) to clear the swinging leg from the ground.

9 conclusion

In this paper a simple model of a biped system is analyzed using nonlinear techniques. The biped dynamics are maintained naturally using limit cycles of nonlinear harmonic oscillators. The use of nonlinear tools has been emphasized in this paper. For instance, the coordination between the two sets of nonlinear dynamics of the legs during the single stance phase has been portrayed. Other forms of coordination are possible [9]. The designed biped system is intended to interact with the environment relying mainly on its intrinsic mechanical properties. This is accomplished by special nonlinear feedback laws which guide the system to find a stable motion pattern with reduced energy expenditure. For example, the biped system only requires a small control effort to maintain a periodic motion for the stance leg by acting in concert with the body mass during the active mode of the single stance phase.

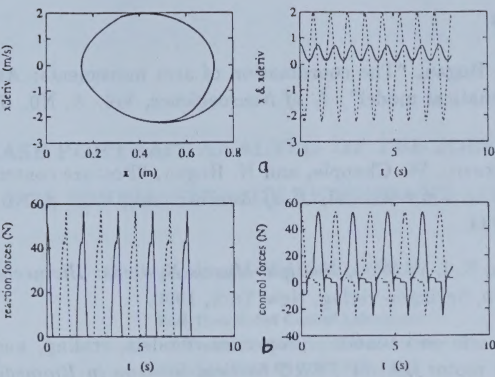


Figure 3: Figures (a) show respectively the limit cycle of the motion and the time history of the trotting gait. The dashed line represents the velocity of the motion. Figures (b) depict the control force history employed to activate the system, as well as the biped reaction forces.

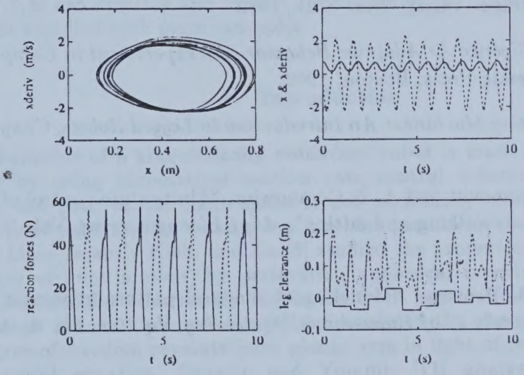


Figure 4: Biped system negotiating a vibrating ground.

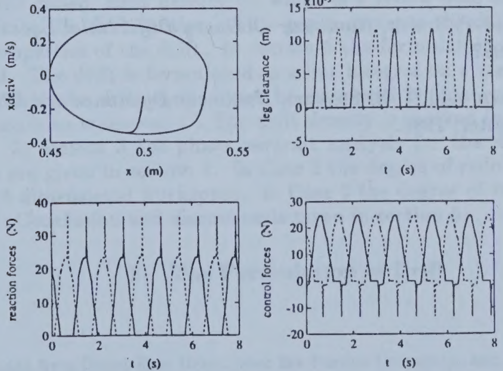


Figure 5: Biped system performing the walking gait.

10 References

[1] T. Flash, and N. Hogan, "The coordination of arm movements: An experimentally confirmed mathematical model", *J. of Neuroscience*, Vol. 5, N0. 7, July 1985, pp. 1688-1703.

[2] E. Bizzi, N. Accornero, W. Chapple, and N. Hogan, "Posture control and trajectory formation during arm movement", *J. of Neuroscience*, Vol. 4, N0. 11, November 1984, pp. 2738-2744.

[3] J. M. Winters and S. L-W Woo, *Multiple Muscle Systems: Biomechanics and Movement Organization*, Springer-Verlag, New York, 1990.

[4] F. E. Zajac, "Muscle and tendon: properties, models, scaling, and application to biomechanics and motor control", *CRC Critical Reviews in Biomedical Engineering*, Vol. 17, 1989, pp. 359-411.

[5] J. J. Denier van der Gon, A. C. C. Coolen, C. J. Erkelins, and H. J. J. Jonker "Self-Organizing neural mechanisms possibly responsible for muscle coordination", *Multiple Muscle Systems: Biomechanics and Movement Organization*, J. M. Winters and S. L-W Woo Eds, Springer-Verlag, New York, 1990.

[6] R. D. Beer, *Intelligence As Adaptive Behavior: An experiment in Computational Neuroethology*, Academic Press, Boston, 1990.

[7] D. J. Todd, *Walking Machines: An Introduction to Legged Robots*, Chapman and Hall, New York, 1985.

[8] H. Hemami, R. Tomovic, and A. Z. Ceranowicz, "Finite state control of planar bipeds with application to walking and sitting", *J. of Bioengineering*, Vol. 2, pp. 477-494, 1978.

[9] J. S. Bay, and H. Hemami, "Modeling of a neural pattern generator with coupled nonlinear oscillators", *J. of Biomedical Engineering*, Vol. 34, No. 4, April 1987, pp. 299-306.

[10] R. Katoh and M. Mori, "Control method of biped locomotion giving asymptotic stability of trajectory", *Automatica* Vol. 20, 1984, pp. 405-414.

[11] D. W. Jordan and P. Smith, *Nonlinear Ordinary Differential Equations*, Clarendron Press, Oxford, 1987.

[12] J. M. T. Thompson and H. B. Stewart, *Nonlinear Dynamics and Chaos*, John Wiley and Sons, Chichester, 1986.

THE PHASE PORTRAIT ANALYSIS OF THE JOINT DRIFT MOTION FOR KINEMATICALLY REDUNDANT ROBOTS

Shengwu Luo and Shaheen Ahmad

Real-Time Robot Control Laboratory
School of Electric Engineering, Purdue University
West Lafayette, IN 47907-0501, USA

Abstract

Redundant robots that are kinematically controlled using Jacobian pseudo-inverses may not have repeatable joint motions. This problem has been observed and analyzed by Klein and Huang[5], Shamir and Yomdin[11]. The mathematical method developed in Luo and Ahmad's paper[7, 8] can describe and predict the drift motion. This paper presents the phase portrait analysis supplied with more examples.

1. Introduction

If the end-effector of a kinematically redundant robot is made to trace a closed path in the workspace by using kinematical motion rate control schemes through the Jacobian pseudo-inverse, the path in the joint space is not necessarily closed. This phenomenon of non-repeatability of joint motion while using Jacobian pseudo-inverse control has been known for a while[5]. More recently Klein and Kee[6] studied the properties of the drift motion (the non-repeatability of joint motion) by performing numerical experiments, they showed that the drift has predictable, numerically stable limits in some situations. Klein and Huang originally observed the drift phenomenon[5] and analyzed the non-repeatability phenomenon for a three degree-of-freedom revolute joint planar arm in light of integrability condition of a Pfaffian differential equation. Shamir and Yomdin [11] analyzed the repeatability of redundant manipulators by the "Lie bracket condition" (LBC). Bay [2] recently proposed the use of Poincare return map method to approximate the drift motion. We proposed a measure of the drift called "drift density"[9] which is a vector field defined on the self-motion manifold to generate the drift motion. This paper presents the drift portrait analysis, that shows the limit properties of the drift. In section 2 the formulation of the drift and the drift density is recalled. The drift is formulated as a line integral on a manifold in the work space. It is then shown that the drift motion can be expressed by the surface integral of the Lie brackets of the Jacobian transpose J^T. The drift density is derived from the derivative of that surface integral. In section 3 the phase portrait analysis for the drift motion is discussed. Two case studies are given in section 4. In Case 1 the degree of redundancy is 1 for the 4-link robot moving in 3-dimensional workspace. In Case 2 the degree of redundancy is 2 for the 4-link planar robot. Conclusion and discussion is given in section 5.

2. The Formulation of Drift

---

† This work was funded by a David Ross Grant from the Purdue University, and by a grant from Shin Meiwa Industries, Japan.

Consider a redundant manipulator with its joint space Q in R^n and its work space X in R^m. The work space velocities and the joint space velocities are related by an m by n Jacobian J :

$$\dot{x} = J\,\dot{q}$$

The pseudo-inverse joint rate control is used to yield the minimum norm joint rates that are computed by

$$\dot{q} = J^+\,\dot{x} \qquad | \quad q(0)=q_0$$

The robot joint position can be solved from the following integral equation when the end-tip of the manipulator traces a parametrized path x(t):

$$q(t) = \int_{t_1}^{t_1+T} J^+(q(t))\dot{x}dt \qquad | \qquad q(0)=q_0 \tag{2.1}$$

This expression enables us to mathematically describe the drift by the following definition.

Definition 1 : The "drift" of the joints during pseudo-inverse control is defined as the difference of the joint position vector undergoing one cycle of path ∂W in the work space, that is a vector-valued line integral of J^+ along a closed path ∂W :

$$dr(\partial W) \triangleq q(t_1+T)-q(t_1) \quad |_{x(t_1+T)=x(t_1)} \quad = \oint_{\partial W} J^+(q)dx \tag{2.2}$$

From this definition we notice that the drift is the difference of the Poincare map which maps $q(t_1)$ to $q(t_1+T)$ on the self-motion manifold. Recall that the drift given by eq(2.2) is a vector, $dr \in IR^n$. Its projection on an unit vector in the direction of the fiber serves as a measure in lower dimension. This motivates following definition of drift density.

Definition 2: Let ker(J) be the null space of the Jacobian J. The projection operator onto ker(J) is denoted by $\mathbf{P}_{ker(J)}$. The "drift density at point q" dd(q) is the projection of drift (area) derivative onto the unit vector at point q in the null space of J. Here the drift derivative is a ratio of the path integral of eq(2.2) on an infinitesimal loop over the area of the surface surrounded by the infinitesimal loop. This is a function of the joint angles q, i.e.,

$$dd(q) \triangleq \mathbf{P}_{ker(J)} \left\{ \lim_{diam D \to 0} \frac{1}{area(D)} dr(\partial W) \right\} \tag{2.3}$$

By the definition and after the calculus on manifold the drift density for m-dimensional workspace can be expressed as the projection of a sum of $\binom{m}{2}$ terms of Lie brackets through following proposition:

Proposition 2: Let $bg=(g_1, \cdots, g_m) : D \rightarrow W$ be a regular transformation from area D in (u,v) plane into workspace W of x, such that $x_i=g_i(u,v), \mid i=1 \cdots m$. Then the drift density defined by eq(2.3) can be computed by

$$dd(q) = \pm(I_n - J^+J) \left\{ \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} [J^+{}_i, J^+{}_j] \frac{\partial(g_i, g_j)}{\partial(u,v)} \right\} \tag{2.4}$$

where $\dfrac{\partial(g_i,g_j)}{\partial(u,v)} = \det \begin{bmatrix} \dfrac{\partial g_i}{\partial u} & \dfrac{\partial g_i}{\partial v} \\ \dfrac{\partial g_j}{\partial u} & \dfrac{\partial g_j}{\partial v} \end{bmatrix}$,

which takes account of the transformation of coordinates . The Lie brackets of J^+ can be expressed by the Lie brackets of J^T if the Jacobian matrix J is of full rank. Thus the drift density can also be expressed in light of the Lie brackets of J^T. Following proposition allows the computation of the drift density.

Proposition 3: The drift density defined by eq(2.3) can be expressed by J^T as a sum of $\binom{m}{2}$ terms of Lie bracket projections:

$$dd = \pm \mathbf{P}_{ker(J)} \left\{ \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} [J^+{}_i, J^+{}_j] \frac{\partial(g_i, g_j)}{\partial(u,v)} \right\}$$

$$= \pm\left[I_n - J^T(JJ^T)^{-1}J\right]\sum_{l_1=1}^{m-1}\sum_{l_2=l_1+1}^{m}\left\{[J^T{}_{l_1}, J^T{}_{l_2}]\,\frac{\partial(g_i,g_j)}{\partial(u,v)}\left[\sum_{i=1}^{m-1}\sum_{j=i+1}^{m}(-1)^{(i+j+l_1+l_2)}(JJ^T)\left(\begin{matrix}(i,j)'\\(l_1,l_2)'\end{matrix}\right)\right]\right\}. \tag{2.5}$$

Where $(JJ^T)\left(\begin{matrix}(i,j)'\\(l_1,l_2)'\end{matrix}\right)$ is the n−2 th-order minor of JJ^T.

Using the concise notation introduced by Gantmacher (p.2, [3]), the minor of A of order p is defined as:

$$A\left(\begin{matrix}\lambda_1\\\lambda_2\end{matrix}\right) = A\left(\begin{matrix}i_1 & i_2 & \cdots & i_p\\ l_1 & l_2 & \cdots & l_p\end{matrix}\right) = \det\begin{bmatrix}a_{i_1l_1} & a_{i_1l_2} & \cdots & a_{i_1l_p}\\ a_{i_2l_1} & a_{i_2l_2} & \cdots & a_{i_2l_p}\\ \cdots & \cdots & \cdots & \cdots\\ a_{i_pl_1} & a_{i_pl_2} & \cdots & a_{i_pl_p}\end{bmatrix}.$$

Where λ_1 and λ_2 are the p-tuples of ordered integers:

$$\lambda_1=(i_1, i_2, \cdots, i_p) \quad | \quad 1\leq i_1<i_2<\cdots<i_p\leq n \quad ;$$

$$\lambda_2=(l_1, l_2, \cdots, l_p) \quad | \quad 1\leq l_1<l_2<\cdots<l_p\leq n \quad .$$

In the equation eq(2.5) λ' denotes the complement of n-2 tuple of ordered integers such that after reordering the n-tuple (λ,λ') becomes (1,2,...,n), so $(i,j)'$ is indeed the (n-2)-tuple obtained after deleting the integer i and j from the n-tuple (1,2,...,n), that is, $(i,j)'=(1,2,\cdots,\hat{i},\cdots,\hat{j},\cdots,n)$, where the symbol $\hat{i}$ means that the integer i is deleted from the n-tuple (1,...,n).
The proofs of Proposition 1, 2 and 3 can be found in [8].

As an example calculation consider the case with m=2 , since the workspace is planar the coordinates transformation is not necessary, eq(2.5) yields the drift density for the path in a two dimensional work space

$$dd\,|_{m=2} = \pm(I_n-J^+J)[J^+{}_1, J^+{}_2]$$

$$= \pm\left[I_n-J^T(JJ^T)^{-1}J\right]\,[J^T{}_1,J^T{}_2]/\det(JJ^T) \tag{2.6}$$

For m=3, we obtain, the drift density for the path in a three dimensional work space,

$$dd\,|_{m=3} = \pm(I_n-J^+J)\Bigg\{ \left[(JJ^T)_{31}-(JJ^T)_{32}+(JJ^T)_{33}\right][J^T{}_1,J^T{}_2]\,\frac{\partial(g_1,g_2)}{\partial(u,v)} + \left[(JJ^T)_{11}-(JJ^T)_{12}+(JJ^T)_{13}\right][J^T{}_2,J^T{}_3]\,\frac{\partial(g_2,g_3)}{\partial(u,v)} - \left[(JJ^T)_{21}-(JJ^T)_{22}+(JJ^T)_{23}\right][J^T{}_3,J^T{}_1]\,\frac{\partial(g_3,g_1)}{\partial(u,v)}\Bigg\}/\det(JJ^T). \tag{2.7}$$

where $(JJ^T)_{ij}$ is the entry in the i-th row and j-th column of the matrix (JJ^T).

3. Phase Portraits of Drift Motions

In this section we will determine the stability behavior exhibited by drift motion. For drift motions, the equilibrium set are the joint configurations at which the drift and hence the drift density is zero, (see eq(2.5)). Let these joint configurations be in the set $\mathscr{E}$ then,

$$\mathscr{E}=\{q\in Q_f \;|\; dd(q)=0;\; x_0=K(q)\}\,. \tag{3.1}$$

Therefore $\mathscr{E}$ comprises of all the points at which drift density $dd(q)=0$. The drift density dd defined in Section 2 describes the drift velocity occurring along the self motion space of x_0. The drift motion caused by this dd can be described by a constrained dynamical system:

$$\dot{q}_f=dd(q_f) \quad | \quad x_0=K(q_f)\,. \tag{3.2}$$

We will use phase portrait method for studying the dynamical system described by eq(3.2).

The phase portrait reveals a "panorama" of the motion trajectories and the stability behavior at every critical point. Nowadays the phase portrait technique is not only a powerful analytical tool but also has a sound theoretical basis. We cite here the formal definitions of the phase portrait and the equivalency of the two phase portraits (p.35,[12]).

Definition 3 Let $F^k(M)$ be the space of the k times differentiable vector field defined on manifold M. Assume that $\Phi \in F^k(M)$. The set $P(\Phi,M)$ of all oriented orbits $\hat{\gamma}(t)$ of Φ is called the *phase portrait* of Φ.

Definition 4 Let M_1, M_2 be manifolds and let $\Phi_i \in F^k(M_i)$, for $i=1,2$. The fields Φ_1 and Φ_2 are said to have *equivalent phase portraits* iff there exists a homeomorphism h of M_1 onto M_2 such that

$$\hat{\gamma} \in P(\Phi_1,M_1) \quad \text{iff} \quad h(\hat{\gamma}) \in P(\Phi_2,M_2)$$

Following diagram illustrates the condition for the equivalency of two phase portraits.

$$\begin{tikzcd}
\hat{\gamma} \arrow[r, "h"] & h(\hat{\gamma}) & \text{Orbits} \\
M_1 \arrow[u, "P(\Phi_1,M_1)"] \arrow[r, "h"] & M_2 \arrow[u, "P(\Phi_2,M_2)\ \text{Phase Portrait}"'] & \text{Manifold}
\end{tikzcd}$$

Let M_1 be the r-dimensional self-motion manifold in IR^n, and M_2 be the subspace in IR^r. We may find a homeomorphism h that maps M_1 to M_2. If we conform to the above condition using the same homeomorphism h for mapping the orbits on M_1 to the orbits on M_2, then we are able to obtain a planar graph that is equivalent to the phase portrait on the manifold in high dimentional space. In our practice we use the arc-length function as h for $r=1$, and use the integration of null space as h for $r=2$.

The basis of the phase portrait method is the Hadamard-Perron Theorem[12], which tells us that in the neighborhood of a hyperbolic fixed point the stable/unstable manifold of the map ϕ is tangent to the stable/unstable subspace of the differential map $d\phi$. Additionally, for non-hyperbolic fixed point, we have the "Center Manifold Theorem"[4], which tells us that there exist (need not be unique) center manifold tangent to the zero-eigenvalue subspace of the differential map $d\phi$. Futhermore, for the stability of the system with empty unstable manifold, the "Reduction Priciple"[4] allows us to reduce the problem to the stability problem of the system constrained on the center manifold. Following those theorems, locally the stability of the drift map (2.2) can be characterized by the phase portrait of the differential map (6.2).

Now we discuss the method for distinguishing the nature of critical points of the system on IR^2 [10].

Consider the planar nonlinear system

$$\dot{x}=f_1(x,y)$$

$$\dot{y}=f_2(x,y) \,. \tag{3.3}$$

We assume the equilibrium point is an isolated critical point of the system (3.3). Let the matrix of linearized system be $A=\frac{\partial(f_1,f_2)}{\partial(x,y)}$. The nature of the critical point is related to the eigenvalues of A. If A is nonsingular (no zero eigenvalue) the critical point of the system (3.3) is hyperbolic. The hyperbolic critical point is either a node, center, focus or saddle, those are common mentioned critical points. The Hartman-Grobman Theorem [1] shows that near a hyperbolic critical point the orbit structure of a nonlinear system is qualitatively the same, (in the sense of being topologically conjugate), as the orbit structure given by the associated linearized system. However when the zero eigenvalue takes place, the critical point is called a non-hyperbolic point. the nature of a non-hyperbolic critical points is more various. We illustrate some examples of them, a saddle-node in Fig.3a, a critical point with an elliptic domain in Fig.3b and a cusp in Fig.3c.

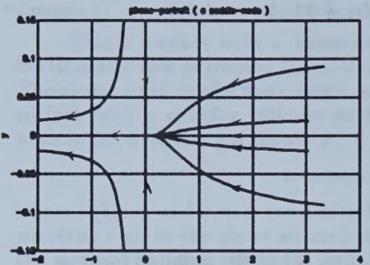


Fig.1a A saddle node

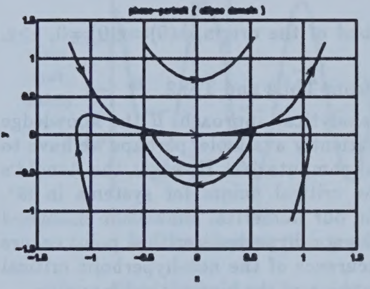


Fig.1b A critical point with ellipse domain.

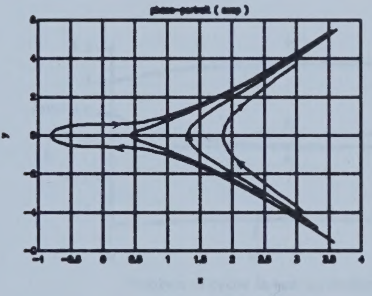


Fig.1c A cusp

The following Arnold's Theorems indicate the method for determining the nature of a non-hyperbolic point of the system (3.3) on $\mathbb{R}^2$. We summarized the theorems with lists of the conditions for the convenience in application. The proofs of those theorems can be found in (p.340, pp.357-362, [1]).

Theorem 1 (Arnold) : [One zero eigen-value case]

Consider the system (3.3) with $\det A=0$ and $\mathrm{tr}A\neq 0$, so it has the nominal form as

$$\begin{aligned}\dot{x}&=p(x,y)\\ \dot{y}&=y+q(x,y)\end{aligned} \qquad A=\begin{bmatrix}\partial p/\partial x & \partial q/\partial x\\ \partial p/\partial y & 1+\partial q/\partial y\end{bmatrix}.$$

Define the function $\phi(x)$ as $y=\phi(x) \quad | \quad \phi(x)+q(x,\phi(x))=0$

and in the neiborgfhood of 0 the function p can be expanded as

$$p(x,\phi(x))=a_m x^m+a_{m-1}x^{m-1}+\ldots\ldots .$$

Then the nature of the critical point can be checked up from Tbl.1 .

m a_m	*odd* >0	*odd* <0	*even*
	node	saddle	saddle-node

Tbl.1 non-hyperbolic critical points (One zero eigenvalue case)

Theorem 2 (Arnold) : [Two zero eigen-values case]

Consider the system (3.3) with $\det A=0$, $\operatorname{tr}A=0$ and $A\neq 0$, so it has the nominal form as

$$\dot{x}=y$$

$$\dot{y}=ax^k(1+h(x))+bx^n y(1+g(x))+y^2R(x,y)$$

where $h(x)$, $g(x)$ and $R(x,y)$ are smooth in a neighborhood of the origin, $h(0)=g(0)=0$, $k\geq 2$, $a\neq 0$ and $n\geq 1$.

Then the nature of the critical point can be checked up from Tbl.2 and Tbl.3.

Above method stated in Arnold's theorems is an analytical approach. If the knowledge of the analytical form of the nonlinear system is not sufficiently available, perhaps we have to count on the phase portrait analysis based on numerical computation. However the Arnold's theorem summarized the classes of the natures of the critical points for systems in $\mathbb{R}^2$. Indeed, we do find some non-hyperbolic critical points in our numerical simulation discussed in next section. As Arnold's Theorem points out that the non-hyperbolic critical point occurs when the linearized system has zero eigenvalues, the occurence of the non-hyperbolic critical point in drift portrait indicates the singularity of the Jacobian of the kinematical mapping.

k even , $k=2m$

$b=0$	$b\neq 0$ $n\geq m$	$b\neq 0$ $n<m$
cusp	cusp	saddle-node

Tbl.2 Non-hyperbolic critical points (Two zeros)

k odd, $k=2m+1$, $\lambda:=b^2+4(m+1)a$

<table>
<tr><td rowspan="4">$a>0$</td><td colspan="7">$a<0$</td></tr>
<tr><td>$b=0$</td><td colspan="2">$b\neq 0$</td><td colspan="4">$b\neq 0$</td></tr>
<tr><td rowspan="2"></td><td colspan="2"></td><td colspan="2">n=even</td><td colspan="2">n=odd</td></tr>
<tr><td>$n>m$</td><td>$n=m$
$\lambda<0$</td><td>$n<m$</td><td>$n=m$
$\lambda\geq 0$</td><td>$n<m$</td><td>$n=m$
$\lambda\geq 0$</td></tr>
<tr><td>saddle</td><td colspan="3">focus or center</td><td colspan="2">node</td><td colspan="2">ellipse domain</td></tr>
</table>

Tbl.3 Non-hyperbolic critical points (Two zeros)

4. Case Study

In this section we give two examples to illustrate the drift motions for different dimension of work space $m=3$ and $m=2$ and for different degree of redundancy $r=1$ and $r=2$. Two things in the simulation are noticeable. Firstly, the phase portrait delivers plenty of knowledge about the drift motion, its velocity, its convergence and the tendency. Secondly, the shape of the drift density and the "actual drift" agree so sufficiently well.

Example 1 : $n=4,\ m=3,\ r=1.$

This is a robot with a three-link, all are unit length, planar manipulator mounted on a horizontally laid prismatic link , thus it forms an open chain robot with 4 degree-of-freedom. The closed path in the work space is chosen to be a curve which is the intersection of a plane surface with a circular cylinder surface. Following is its parametric equations of the planar path in $\mathbb{R}^3$ with one parameter ϕ.

$$x_1=a+r\cos\phi \qquad x_2=b+r\sin\phi \qquad x_3=(x_1+cx_2)-d.$$

where a,b,c,d and r are constants for determining the position and size of the path. The constant c gives the plane an inclination to the x_1,x_2 plane, and r determines the loop size of the path. The other constant can be determined by the initial value of joints q. In this simulation we chose $c=2$ and $r=0.2$.

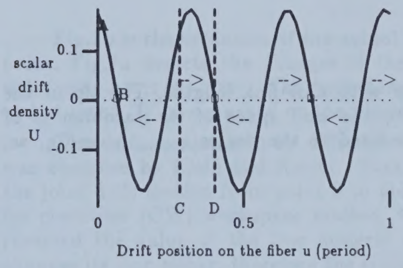


Fig.2a Phase portrait (n=4,m=3,r=1)

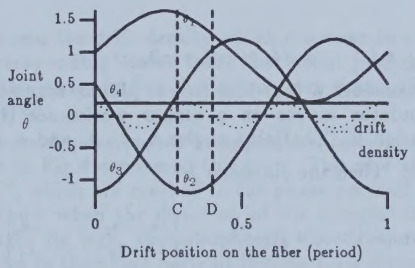


Fig.2b Joint motion

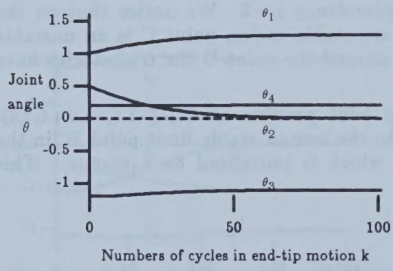


Fig.3a motion of states (from A to B)

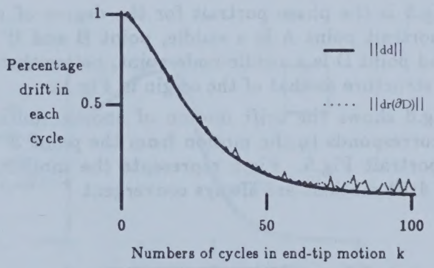


Fig.3b convergence of the drift

In Fig.2a the horizontal axis u is the arc-length that comes from the integral of tangent vector t°, and the vertical axis takes on the inner product $U=\langle dd, t^\circ\rangle$. The stable equilibrium points are marked by the symbol □ in the figure. Fig.2b is the graph of the homeomorphism $u=u(q_f)$. Fig.3 is the evolution of the convergent drift motion, which corresponds to the motion from the leftmost point to the first stable limit point in the phase portrait Fig.2. Fig.4 shows the drift motion of another initial joint position at $\theta(0)=(1.6,-1.16,0.3,0.2)$. which corresponds to the motion from the point p to the second stable limit point s in the phase portrait Fig.2.

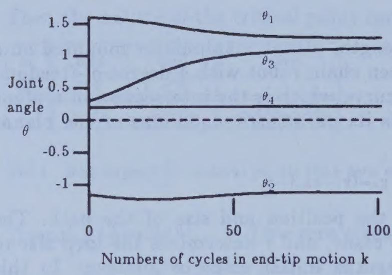


Fig.4a motion of states (from C to D)

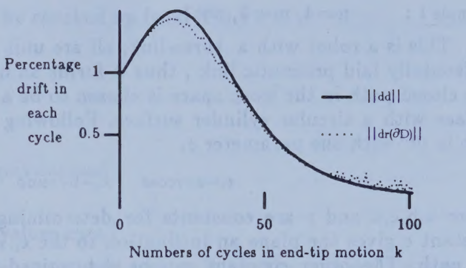


Fig.4b convergence of the drift

Example 2 : n=4, m=2, r=2.

Consider a four-link (n=4) planar manipulator with unit link lengths. The tip of the manipulator moves on a planar workspace (m=2), the null space of its Jacobian is of dimension 2. The joint angles θ_1, θ_2, θ_3 and θ_4 are related to the angles q_1, q_2, q_3 and q_4 as, $q_k=\sum_{i=1}^{k}\theta_i$. Then the Jacobian J

with respect to q is given by $J=\begin{bmatrix} -s_1 & -s_2 & \cdots & -s_n \\ c_1 & c_2 & \cdots & c_n \end{bmatrix}$

with $s_i=\sin(q_i)$ and $c_i=\cos(q_i)$. The actual drift is found by integrating $\dot{q}$ over the closed loop in the workspace by eq(2.1). Gear's ordinary differential equation solver is used for the integration.

Fig.5 is the phase portrait for the degree of redundancy r=2. We notice that on the phase portrait point A is a saddle, point B and B' are stable nodes, point C is an unstable node and point D is a saddle-node point, notice that around the point D the trajectories have similar structure as that of the origin in Fig.1a.

Fig.6 shows the drift motion of another initial joint position at $\theta(0)=(-1.5,-1.0,1.0,1.5)$. which corresponds to the motion from the point S' to the second stable limit point B' in the phase portrait Fig.5. Fig.6 represents the motion, which is initialized by $\theta_2(0)=\theta_3(0)$. This kind of drift motions are always convergent.

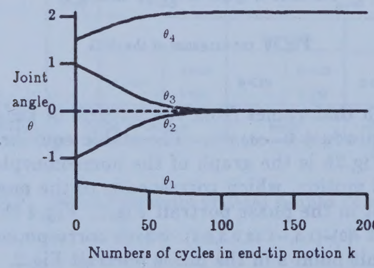


Fig.6a motion of states (from S' to B')

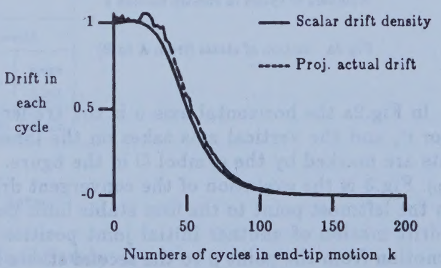


Fig.6b Convergence of the drift

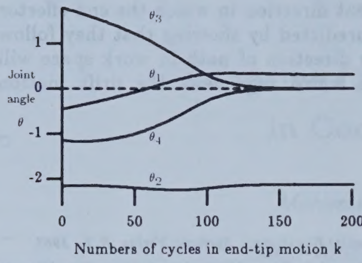


Fig.7a motion of states (from S to B)

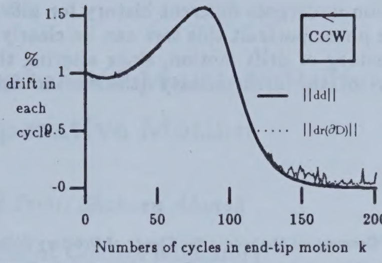


Fig.7b convergence of the drift

Fig.7b is the evolution of the actual drift and the drift density which converge to a limit point. Fig.7a depicts the changes of the corresponding states from the initial joint angles $\theta(0)=(-0.6,-2.07,1.96,-1.24)$. The trajectory of the states in Fig.7b started from point "S" and end at point "B" are marked on the phase portrait.

The drift motion will be different if the workspace moves in opposite direction, which was observed by Klein and Kee[6]. Next plot in Fig.8 confirmed this fact. This plot showes the joint drift motion from point S to point C', which are marked in the phase portrait Fig.5 for clockwise (CW) workspace motion. We know when the direction of the integral loop is reversed the value of the line integral changes its sign, correspondingly the drift density changes its sign either, therefore the trajectories in the phase portrait reverse their directions. All the sources become the drains and vis-versa. The point C' now is a stable node for CW motion. The drift motion starts from point S will converge to point B for CCW path, and will converge to point C' for CW path, that clears the reason why different winding direction produces different drift motion.

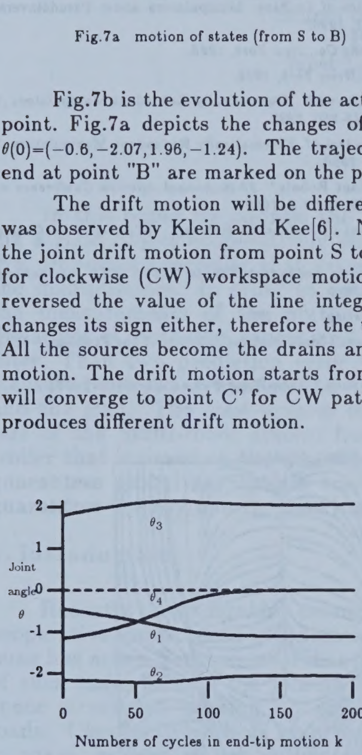


Fig.8a motion of states (from S to C')

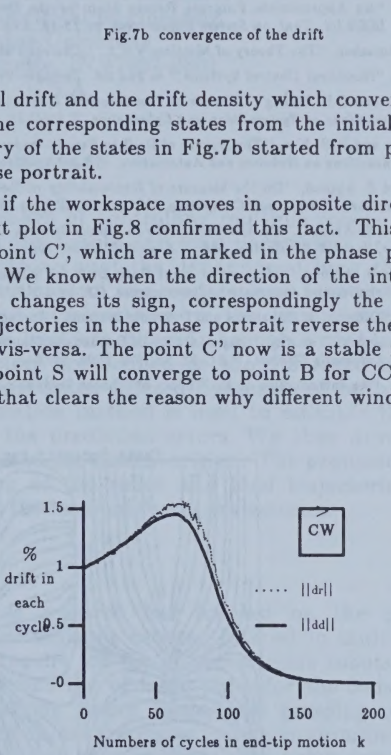


Fig.8b convergence of the drift

5. Conclusion

The above discussions are now summarized in the below.

1) The drift density we proposed gives a good measure of the drift motion of the redundant robot and explains well the limit phenomenon of the drift.

2) The mathematical method presented in this paper provides useful tools to analyze the drift in higher dimensional problems.

3) This paper emphasizes on the higher dimensions, by that we mean the dimension of the work space m>2 and/or the degree of redundancy r>1. Our simulations are chosen for the case of m=3 or r=2. The phase portrait of the drift motion in the simulations has indicated the nature of the critical points. In a certain neighborhood the trajectories of the drift motion do converge to the predicted stable zero drift joint configuration. As a experimental fact the

joint configuration undergoes different history for different direction in which the end-effector runs. Using the phase portrait this fact can be clearly predicted by showing that they follow a different trajectory of drift motion, since altering the direction of path in work space will change the sign of the drift density (the vector field which represents the drift motion dynamics).

□

References

1. V.I. Arnold, "Geometrical Methods in the Theory of Ordinary Differential Equations," *Springer-Verlag, N.Y., 1983.*
2. J.S. Bay, "An Approximate Poincare Return Map for the Dynamics of Drifting Manipulators under Pseudoinverse Control," *IEEE Int. Conf. on System Engineering, pp.13-16, Aug. 9-11, 1990.*
3. F.R. Gantmacher, "The Theory of Matrices Vol.1,," *Chelsea Publishing Co., New York, 1960.*
4. A. Isidori, "Nonlinear Control Systems," in *2nd Ed. Springer-Verlag,Newq York, 1989.*
5. C.A. Klein and C.H. Huang , "Review of Pseudoinverse Control for Use with Kinematically Redundant Manipulators ," *IEEE Transactions on System, Man and Cybernetics, V.SMC-13, pp.245-250, 1983.*
6. C.A. Klein and K.B. Kee, "The Nature of Drift in Pseudo-inverse Control of Kinematically Redundant Manipulators ," *IEEE Transactions on Robotics and Automation, V.RA-5. pp231-234, 1989.*
7. S. Luo and S. Ahmad, "On the Measure of Repeatability of Redundant Robots," *28-th Annual Allerton Conference on Communication, Control, and Computing, (Monticello, Ill.), pp.204-205, Oct.3-5, 1990.*
8. S. Luo and S. Ahmad, "On the Drift Rate in Pseudo-Inverse Control of Redundant Robots," *1991 American Control Conference, June 26-28, Boston, MA, V.2, pp.2021-2026, 1991..*
9. S. Luo and S. Ahmad, "Measure of Joint Path Drift for Kinematically Redundant Robots," *1991 IEEE International Conf. on Robotics and Automation, Sancramento, CA. pp.1163-1168, 1991.*
10. L. Perko, "Differential Equations and Dynamic Systems," *Springer-Verlag, N.Y., 1991.*
11. T. Shamir and Y. Yomdin,, "Repeatability of Redundant Manipulators: Mathematical Solution of the Problem," *IEEE Trans. on Automatic Control, V.AC-33, pp.1004-1009, 1988.*
12. W. Szlenk, "An Introduction to the Theory of Smooth Dynamical Systems," *John Wiley & Sons, Chichester, 1984.*

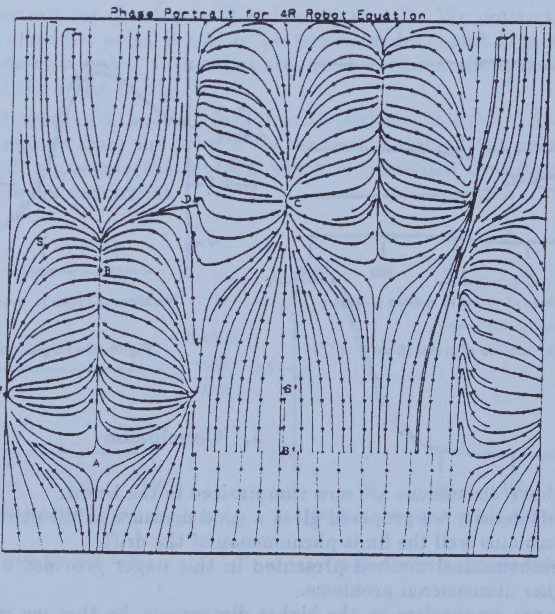


Fig..5 Phase Portrait (n=4, m=2, r=2)

Adaptive Control of Multiple Robots in Cooperative Motion

Mohamed Zribi, Shaheen Ahmad

Real-time Robot Control laboratory
School of Electrical Engineering
Purdue University
West Lafayette, IN 47907-0501
U.S.A

Abstract

In this paper we address the problem of controlling multiple robots manipulating a rigid object cooperatively when the robots and load parameters are uncertain. We propose a controller that takes into account the dynamics of both the load and the manipulators. In order to design a control and update laws that do not require the measurements of the of the joint accelerations or the load acceleration, the dynamics of both the robots and the load are filtered through a stable first order filter. Then two prediction error vectors are defined as the difference between the measured filtered dynamics and the predicted filtered dynamics of both the robots and the load. The least-squares estimation method is used to estimate the parameters of the multi-robot system from the prediction errors. We then develop a controller that is based on the cancellation of the nonlinearities. The proposed controller guarantees global asymptotic tracking of the robot and load trajectories and also guarantees the asymptotic tracking of the internal forces trajectories.

1. Introduction

Recently considerable amount of research has focused on the problem of cooperative control and coordination of multiple robots. Interest in multi-robot systems has arisen because several tasks require the use of two or more robots. Examples of such tasks include the joining and securing of large pipes for the construction of space structures, picking up and carrying heavy loads, and grasping odd shaped loads. Cooperative robots may be used in hazardous or unsafe environments such as in space, in deep waters and in radioactive environments. By using more than one robot the manipulation capability and the workspace of the system are further increased. However the multi-robot systems are more difficult to control than single robots. Additional problems arise as the parameters of the robots and the manipulated load may not be known exactly.

Ortega and Spong [6] presented an excellent summary of several recent papers on adaptive control for rigid single robots. Craig et. al. [2] proposed an adaptive control law which is basically a modification of the computed torque control scheme. Slotine and Li [8] exploited the structural properties of the rigid manipulator dynamics to design an adaptive controller capable of general trajectory tracking control. They tested their design on a two degree of freedom semi-direct drive robot. They experimentally showed that the parameters of the robot which were assumed to be unknown initially can be estimated within a very short time. Bayard and Wen [1] introduced a class of asymptotically stable adaptive control laws for robotic manipulators. They used the property that the dynamics are linear with respect to the unknown parameters of the manipulator. They proposed an energy like Lyapunov function that retains the nonlinear character and structure of the dynamics. The authors proved that their approach leads to asymptotically stable adaptive systems. Their approach does not require the convergence of the parameter estimates, invertibility of the mass matrix estimate or the measurement of joint accelerations. Sadegh and Horowitz [7] presented a control scheme that uses the desired trajectory outputs to update the parameters and the controller. They also address the robustness properties of the proposed scheme. Middleton and Goodwin [5] examined the use of predictive adaptive control laws to a rigid link manipulator. They used a computed torque controller in conjunction with linear estimation techniques to prove the global convergence of the adaptive system.

Two important developments for the adaptive control of single robots are relevant to our work on the control of multiple robots. The first development was the use of linearity of the uncertain parameters of the robot with respect to the dynamics of the robot. The second development is the use of the predictive adaptive control to design control laws for rigid robots. We will use these two developments to design an adaptive controller for the multi-robot system.

Few adaptive control schemes for cooperative multiple robots manipulating a common load have been proposed. Walker et al. [10] developed an algorithm for the control of two robots handling a load of unknown mass. Their controller achieves global convergence of the load motion and the internal forces to their respective desired trajectories. Their controller basically amounts to an open loop controller for the internal forces and a closed loop controller for the positions. The attractive feature of the proposed control law is that the computational algorithm is the same for the two manipulators. Zribi and Ahmad [11] proposed an adaptive controller for the multi-robot system manipulating a rigid object cooperatively. Their controller takes into account the dynamics of the manipulators and the load and does not require feedback of joint acceleration or the inversion of the inertia or the Jacobian matrices. They also considered the effects of bounded disturbances on the system. A control law which guarantees the convergence of the tracking error to a bounded set is also given.

In this paper we address the problem of controlling multiple robots handling an object cooperatively. In section 2 we develop the dynamic models of the manipulators and the load. In section 3 we exploit the linearity of the dynamics with respect to the unknown parameters to derive the parameters update laws. The dynamics of both the manipulators and the load are filtered through a first order stable filter so that the control and update laws do not require the measurements of the accelerations. We define two prediction error vectors for the load and the manipulators. These prediction errors are the difference between the measured filtered dynamics and the predicted filtered dynamics. In section 4 we use these prediction errors with a Least-squares estimation method to estimate the parameters of the multi-robot system. We then develop a controller which guarantees the asymptotic convergence of the object motion and the internal forces exerted by the manipulators on the load to their respective desired trajectories.

2. Cooperative Multi-Robot System Model

2.1 Dynamics Model

The general dynamic model for k cooperative multi-robot system has been investigated thoroughly in the literature, and is also described in the below for completeness. We first start by stating few assumptions that will be used in the subsequent derivation.

Assumptions:

(1) The manipulators are rigidly grasping the load.

(2) The number of joints of the ith robot, n_i, is equal to six and all the manipulators are non redundant.

(3) All the manipulators can independently move along each axis and therefore all the robots are nonsingular along the desired trajectories.

The dynamic equation of the ith manipulator in cooperative manipulation is given as:

$$H_i(q_i)\ddot{q}_i + N_{1i}(q_i,\dot{q}_i) = \tau_i - J_i(q_i)^T F_i \qquad i=1,\ \ldots,\ k \tag{1}$$

where, $q_i \in R^{n_i}$ is the vector of joint displacements, and n_i is the number of joints of the ith robot. The inertia matrix of the ith robot is $H_i(q_i) \in R^{n_i \times n_i}$, this is a positive definite and symmetric matrix. The vector of centrifugal, Coriolis and gravity forces is $N_{1i}(q_i,\dot{q}_i) \in R^{n_i}$; the manipulator Jacobian is $J_i(q_i) \in R^{6\times n_i}$. The control input torque for the ith robot is $\tau_i \in R^{n_i}$. The forces/moments applied by the ith manipulator on the object at the point of contact is $F_i \in R^6$ and it can be written in terms of the contact forces $f_i \in R^3$ and contact moments $\eta_i \in R^3$, such that $F_i = \begin{bmatrix} f_i^T & \eta_i^T \end{bmatrix}^T$ for $i = 1,\ldots,k$.

Now we will group the dynamics of the k robots to get,

$$H(q)\ddot{q} + N_1(q,\dot{q}) = \tau - J^T(q)F \tag{2}$$

Let $n = \sum_{i=1}^{i=k} n_i = 6k$ where n_i is the number of joints of the ith robot. Then it should be noted that $H \in R^{n\times n}$, $N_1 \in R^n$, $\tau \in R^n$, $J \in R^{n\times n}$, $F \in R^n$, $q \in R^n$. Here $H(q)$ is the block diagonal matrix whose diagonal elements are $H_i(q_i)$, (i.e $H = diag(H_i)$), also $J(q)$ is the block diagonal matrix whose diagonal elements are $J_i(q_i)$, (i.e $J = diag(J_i)$). Further,

$$N_1(q,\dot{q}) = \begin{bmatrix} N_{11}(q_1,\dot{q}_1) \\ \cdot \\ \cdot \\ \cdot \\ N_{1k}(q_k,\dot{q}_k) \end{bmatrix}, \quad q = \begin{bmatrix} q_1 \\ \cdot \\ \cdot \\ \cdot \\ q_k \end{bmatrix}, \quad \tau = \begin{bmatrix} \tau_1 \\ \cdot \\ \cdot \\ \cdot \\ \tau_k \end{bmatrix} \quad \text{and} \quad F = \begin{bmatrix} F_1 \\ \cdot \\ \cdot \\ \cdot \\ F_k \end{bmatrix} \tag{3}$$

A property that will be used in the subsequent development is the linearity of $H(q)$ and $N_1(q,\dot{q})$ with respect to the manipulators dynamic parameters P_1. We have,

$$H(q)\ddot{q} + N_1(q,\dot{q}) = Y_1(q,\dot{q},\ddot{q})P_1 \tag{4}$$

where $P_1 \in R^{r_1}$ is a vector of r_1 robot parameters which are constants for the given manipulators. These parameters will be estimated by the proposed adaptive scheme. The regressor matrix $Y_1(q,\dot{q},\ddot{q}) \in R^{n\times r_1}$ represents the structure of the robots dynamics, hence its elements are combinations of the elements of the inertia matrix and the centrifugal, Coriolis and the gravity vector.

The equations of motion of the load are obtained from the Newton-Euler mechanics,

$$M\ddot{x} + Mg = \sum_{i=1}^{i=k} f_i \tag{5a}$$

$$I\dot{\omega} + \omega\times(I\omega) = \sum_{i=1}^{i=k} (\eta_i + r_i\times f_i) \tag{5b}$$

Where the position of the center of mass of the object expressed in world coordinate frame is $x \in R^3$. The rotational velocity of the center of mass of the object in world coordinate frame is $\omega \in R^3$, and the gravitational forces acting on the object is $g(x) \in R^3$. The mass matrix $M \in R^{3\times 3}$ is a diagonal matrix whose diagonal elements are the mass of the load; $I \in R^{3\times 3}$ is the inertia matrix of the load. The position of the end effector of the ith manipulator with respect to the object center of mass, expressed in the world coordinate frame, is $r_i \in R^3$.

The motion of the load expressed by equations (5a) and (5b) can be re-written as:

$$S(z)\ddot{z} + N_2(z, \dot{z}) = GF \tag{6}$$

Where $G \in R^{6 \times n}$ is the grasp matrix, and it is defined as

$$G = \begin{bmatrix} T_1 & T_2 & \cdots & T_k \end{bmatrix} \tag{7}$$

The matrix $T_i \in R^{6 \times 6}$ is obtained from equations (5a) and (5b) and it is given as

$$T_i = \begin{bmatrix} I_{3 \times 3} & 0 \\ \Omega_i(r_i) & I_{3 \times 3} \end{bmatrix} \quad \text{and} \quad \Omega_i(r_i) = \begin{bmatrix} 0 & -r_{iz} & r_{iy} \\ r_{iz} & 0 & -r_{ix} \\ -r_{iy} & r_{ix} & 0 \end{bmatrix}$$

where $I_{3 \times 3}$ is the 3 by 3 identity matrix, and $r_i = [\, r_{ix}, r_{iy}, r_{iz}]^T$. Also the matrices $S \in R^{6 \times 6}$ and $N_2 \in R^{6 \times 6}$ are defined such that,

$$S = \begin{bmatrix} M & 0 \\ 0 & I \end{bmatrix} \quad \text{and} \quad N_2 = \begin{bmatrix} Mg \\ \omega \times (I\omega) \end{bmatrix}$$

The angular velocity of the center of mass of the object expressed in world coordinate frame is $v \in R^3$. The position and orientation vector of the center of mass of the object expressed in world coordinate frame is $z \in R^6$ and the velocity vector is $\dot{z} = \begin{bmatrix} v^T & \omega^T \end{bmatrix}^T \in R^6$. Notice that $\dot{z} = \begin{bmatrix} \dot{x}^T & \omega^T \end{bmatrix}^T$.

Another property that will be used in the development deals with the linearity in the load parameters,

$$S(z)\ddot{z} + N_2(z, \dot{z}) = Y_2(z, \dot{z}, \ddot{z})P_2 \tag{8}$$

where $P_2 \in R^{r_2}$ is a vector of r_2 load parameters which are constants for a given load. These parameters will be estimated by the proposed adaptive scheme. The regressor matrix $Y_2(z, \dot{z}, \ddot{z}) \in R^{6 \times r_2}$ represents the structure of the load dynamics.

2.2 Kinematic Model

We might be interested in controlling the manipulators in some predefined Cartesian task space such that:

$$z_i = K_i(q_i) \qquad i = 1, \ldots, k \tag{9}$$

where $K_i(.) : R^{n_i} \rightarrow R^6$ is the transformation from the joint angle space of q_i to the task space containing z_i, and $z_i \in R^6$ is the position and orientation of the point of contact of the ith manipulator, with the load, expressed in the world coordinate frame. Notice that the angular velocity of the point of contact of the ith manipulator, with the load, expressed in world coordinate frame is $v_i \in R^3$, the rotational velocity of the point of contact of the ith manipulator, with the load, expressed in world coordinate frame is $\omega_i \in R^3$, $\dot{z}_i = \begin{bmatrix} v_i^T & \omega_i^T \end{bmatrix}^T \in R^6$.

If we differentiate equation (9) with respect to time, and if we define $J_i(q_i)$ to be the differential map of the q_i space to z_i space (i.e. $J_i = \dfrac{\partial K_i}{\partial q_i}$), then we can write

$$\dot{z}_i = J_i(q_i)\dot{q}_i \qquad i = 1, \ldots, k \tag{10}$$

If these equations are stacked into a single vector by forming the J_i into a block diagonal matrix, and concatenating the $\dot{q}_i$'s into one vector $\dot{q}$, we get,

$$v_c = J\dot{q} \tag{11}$$

where $v_c = \begin{bmatrix} \dot{z}_1^T & \dot{z}_2^T & \ldots . \dot{z}_k^T \end{bmatrix}^T$ is the vector velocity of the contact points.
If we write F_o as the total force acting on the center of mass of the object, then F_o

correspond to the left hand side of equation (6), hence equation (6) can be written as,

$$F_o = GF \tag{12}$$

Now from the duality between the forces and the velocities, we can write

$$G^T\dot{z} = v_c \tag{13}$$

where $\dot{z}$ is the velocity of the center of mass of the object.
Thus for the k robots system, we combine equations (11) and (13) to get,

$$G^T\dot{z} = J\dot{q} \tag{14}$$

where G is the grasp matrix for the multi-robot system, and J is the Jacobian of the system. Now if we differentiate the above equation, we get,

$$\ddot{q} = J^{-1}G^T\ddot{z} - J^{-1}\dot{J}\dot{q} \tag{15}$$

2.3 Definition of Position and Internal Forces Errors

We require the load to follow a predefined twice differentiable trajectory $z_d \in C^2$ hence the trajectory tracking error of the load, $e \in R^6$, is:

$$e = z - z_d \tag{16}$$

The end effector force of the ith manipulator, F_i, can be decomposed into two forces, the motion force and the internal force. The internal force does not cause any motion of the load. However we must control this force in order to prevent excessive compressive or expansive forces being applied to the load.

We can solve for F from equation (6), we get,

$$F = G^+(S\ddot{z} + N_2) + F_I$$

where F_I denotes the internal grasping forces, and $G^+ = G^T(GG^T)^{-1}$. Let $F_{I,d}$ represent the desired internal grasping forces. Thus the internal grasping forces error is,

$$e_F = F_I - F_{I,d} \tag{17}$$

Notice that *the internal forces and the internal forces errors* do not contribute to the motion of the load.

3. Preliminary Definitions of the Multi-robot System

Let $\hat{P}_1$ be the vector of estimates of the parameters of the robots, then the error vector in the estimates of the robots parameters is $\tilde{P}_1 = \hat{P}_1 - P_1$. Similarly, we can write the parameter estimation error vector for the load as $\tilde{P}_2 = \hat{P}_2 - P_2$. Similarly to equation (4), we can write,

$$\hat{H}\ddot{q} + \hat{N}_1 = Y_1\hat{P}_1 \tag{18}$$

where $\hat{H}$ is the estimate of the inertia matrix $H(q)$ and $\hat{N}_1$ is the estimate of the Coriolis, centrifugal and the gravity vector. Notice that $\tilde{H}\ddot{q} + \tilde{N}_1 = Y_1\tilde{P}_1$, where $\tilde{H}$ is the error in the inertia matrix, and $\tilde{N}_1$ is the error in the Coriolis, centrifugal and gravity vector.

Similarly to equation (8), we can write,

$$\hat{S}\ddot{z} + \hat{N}_2 = Y_2\hat{P}_2 \tag{19}$$

where $\hat{S}$ is the estimate of S and $\hat{N}_2$ is the estimate of N_2.

We want to design control and parameter update laws that does not require the measurements of the acceleration vectors $\ddot{q}$ and $\ddot{z}$, thus we will filter the dynamics of both the multi-robot system and the dynamics of the load. The idea of filtering the dynamics to get rid of the measurements of the acceleration vectors was motivated by the work of Middleton and Goodwin [5]. We multiply both sides of the equation we want to filter by $\lambda_f/(D + \lambda_f)$, where λ_f is a known positive constant and D is the

Laplace operator [15].

Equations (2) and (4) can be combined so that we can write,

$$Y_1 P_1 = \tau - J^T F \tag{20}$$

Now we filter equation (20), to get,

$$Y_{1f} P_1 = \tau_f - (J^T F)_f \tag{21}$$

where,

$$(.)_f \triangleq (\frac{\lambda_f}{D + \lambda_f})(.) \tag{22}$$

The prediction error for the manipulators parameters will be defined as,

$$e_{p_1} \triangleq \tau_f - (J^T F)_f - Y_{1f} \hat{P}_1 = -Y_{1f} \tilde{P}_1 \tag{23}$$

Similarly filtering the equation for the load, we get,

$$Y_{2f} P_2 = (GF)_f \tag{24}$$

The prediction error for the load parameters will be defined as,

$$e_{p_2} \triangleq (GF)_f - Y_{2f} \hat{P}_2 = -Y_{2f} \tilde{P}_2 \tag{25}$$

Finally we will state two lemmas which will be used in the derivation of the control law.

Lemma 1:
If V_1 is a matrix and V_2 is a vector, both of appropriate dimensions, then we have

$$\frac{D + \lambda_f}{\lambda_f}(V_1 V_2) = \frac{1}{\lambda_f} \dot{V}_1 V_2 + V_1 \frac{D + \lambda_f}{\lambda_f}(V_2)$$

the proof for lemma 1 is obvious and follows directly from the definition.

Lemma 2:
Let

$$e = H(s) r$$

where $H(s)$ is an $n \times m$ strictly proper, exponentially stable transfer function. Then $r \in L_2$ implies that $e \in L_2 \cap L_\infty$, $\dot{e} \in L_2$, e is continuous, and $e \longrightarrow 0$ as $t \longrightarrow \infty$. If, in addition, $r \longrightarrow 0$ as $t \longrightarrow \infty$, then $\dot{e} \longrightarrow 0$.

Lemma 2 is taken from Desoer and Vidyasagar [3], see also Ortega and Spong [6].

4. Parameter Estimation and Control law Design

Theorem 1: Parameter Estimation

The below update laws guarantee that $\hat{P}_1$, $\hat{P}_2$ are bounded, and that e_{p_1}, $e_{p_2} \in L_2$ regardless of the control law.

$$\dot{\hat{P}}_1 = \Gamma_1(t) Y_{1f}^T e_{p_1} \tag{26}$$

$$\dot{\Gamma}_1(t) = -\Gamma_1(t) Y_{1f}^T Y_{1f} \Gamma_1(t) \tag{27}$$

and the load parameters are updated as,

$$\dot{\hat{P}}_2 = \Gamma_2(t) Y_{2f}^T e_{p_2} \tag{28}$$

$$\dot{\Gamma}_2(t) = -\Gamma_2(t) Y_{2f}^T Y_{2f} \Gamma_2(t) \tag{29}$$

where $\Gamma_1(t) \in R^{r_1 \times r_1}$, $\Gamma_2(t) \in R^{r_2 \times r_2}$ and $\Gamma_1(t_0)$ and $\Gamma_2(t_0)$ are symmetric positive definite matrices.

Proof:
Consider the following Lyapunov function candidate:

$$V = \tilde{P}_1^T \Gamma_1(t)^{-1} \tilde{P}_1 + \tilde{P}_2^T \Gamma_2(t)^{-1} \tilde{P}_2 \tag{30}$$

If we differentiate V with respect to time, we get,

$$\dot{V} = 2\tilde{P}_1^T \Gamma_1(t)^{-1} \dot{\tilde{P}}_1 + \tilde{P}_1^T \dot{\Gamma}_1(t)^{-1} \tilde{P}_1 + 2\tilde{P}_2^T \Gamma_2(t)^{-1} \dot{\tilde{P}}_2 + \tilde{P}_2^T \dot{\Gamma}_2(t)^{-1} \tilde{P}_2 \tag{31}$$

and noting that $\frac{d}{dt}(\Gamma_i^{-1}\Gamma_i) = 0$ for i=1,2, we get,

$$\dot{\Gamma}_i^{-1} = -\Gamma_i^{-1}\dot{\Gamma}_i\Gamma_i^{-1} = \Gamma_i^{-1}(\Gamma_i Y_{\mathrm{if}}^T Y_{\mathrm{if}} \Gamma_i)\Gamma_i^{-1} = Y_{\mathrm{if}}^T Y_{\mathrm{if}} \tag{32}$$

Thus,

$$\begin{aligned}\dot{V} &= 2\tilde{P}_1^T \Gamma_1^{-1}\Gamma_1 Y_{1f}^T e_{p_1} + \tilde{P}_1^T Y_{1f}^T Y_{1f}\tilde{P}_1 + 2\tilde{P}_2^T \Gamma_2^{-1}\Gamma_2 Y_{2f}^T e_{p_2} + \tilde{P}_2^T Y_{2f}^T Y_{2f}\tilde{P}_2 \\ &= -2e_{p_1}^T e_{p_1} + e_{p_1}^T e_{p_1} - 2e_{p_2}^T e_{p_2} + e_{p_2}^T e_{p_2}\end{aligned} \tag{33}$$

Hence we have,

$$\dot{V} = -e_{p_1}^T e_{p_1} - e_{p_2}^T e_{p_2} = -\tilde{P}_1^T Y_{1f}^T Y_{1f}\tilde{P}_1 - \tilde{P}_2^T Y_{2f}^T Y_{2f}\tilde{P}_2 \tag{34}$$

Therefore $\dot{V} \leq 0$ and the estimator is always stable. Thus $\hat{P}_1$, $\hat{P}_2$ are bounded, and e_{p_1} , $e_{p_2} \in L_2$. This concludes the proof of theorem 1.

In the following we will assume that the parameters $\hat{P}_1$ and $\hat{P}_2$ are in some convex region C_r such that $\hat{H}$ and $\hat{S}$ are positive definite matrices. (i.e. $\hat{P}_1$ and $\hat{P}_2 \in C_r$ / $\hat{H}$ and $\hat{S} > 0$).

Theorem 2: Control Law
The adaptive control law given below, equations (35) - (39), used in conjunction with the update laws given in Theorem 1 guarantees the global convergence of the cooperative multi-robot system tracking errors (i.e. $e \longrightarrow 0$ and $e_F \longrightarrow 0$ as $t \longrightarrow \infty$).

$$\tau = \tau_1 + \tau_2 + \tau_3 + \tau_4 \tag{35}$$

where τ_1, τ_2, τ_3 and τ_4 are defined as

$$\tau_1 = J^T(F_{I,d} - K_I \textstyle\int e_F) \tag{36}$$

$$\tau_2 = -\hat{H}_T(-\ddot{z}_d + K_v\dot{e} + K_p e) + (-\hat{H}J^{-1}\dot{J}\dot{q} + \hat{N}_1 + J^T G^+ \hat{N}_2) \tag{37}$$

$$\tau_3 = -\frac{1}{\lambda_f} J^T \dot{J}^{-T} e_{p_1} - \frac{1}{\lambda_f} J^T G^+ \hat{A}\dot{\hat{A}}^{-1}(GJ^{-T}e_{p_1} + e_{p_2}) \tag{38}$$

$$\tau_4 = \frac{1}{\lambda_f} J^T G^+ Y_{2f}\dot{\hat{P}}_2 + \frac{1}{\lambda_f} Y_{1f}\dot{\hat{P}}_1 \tag{39}$$

$$\hat{H}_T \triangleq \hat{H}J^{-1}G^T + J^T G^+ \hat{S} \tag{40}$$

$$\hat{A} \triangleq GJ^{-T}\hat{H}_T = GJ^{-T}\hat{H}J^{-1}G^T + \hat{S} \tag{41}$$

and the adaptation laws are given by equations (26) - (29).

The matrices K_v, K_p, are defined to be constant symmetric positive definite matrices; K_I is a constant symmetric positive semidefinite matrix. These matrices are design parameters. It should be noted that the choice of τ_1 was motivated by the work of Hsu et al. [4].

Proof:
Applying the inverse filter to equation (25), we get,

$$\frac{D + \lambda_f}{\lambda_f}(e_{p_2}) = GF - Y_2\hat{P}_2 - \frac{1}{\lambda_f} Y_{2f}\dot{\hat{P}}_2 \tag{42}$$

From equation (42) we can solve for F, to get,

$$F = G^{+}[\frac{D+\lambda_f}{\lambda_f}(e_{p_2}) + Y_2\hat{P}_2 + \frac{1}{\lambda_f}Y_{2f}\dot{\hat{P}}_2] + F_I \tag{43}$$

where F_I is the internal grasping forces, and $G^{+} = G^T(GG^T)^{-1}$.

Also applying the inverse filter to equation (23), we get,

$$\frac{D+\lambda_f}{\lambda_f}(e_{p_1}) = \tau - J^TF - Y_1\hat{P}_1 - \frac{1}{\lambda_f}Y_{1f}\dot{\hat{P}}_1 \tag{44}$$

Now combining equations (43) and (44), we get,

$$\begin{aligned}\frac{D+\lambda_f}{\lambda_f}(e_{p_1}) + J^TG^{+}\frac{D+\lambda_f}{\lambda_f}(e_{p_2}) &= \tau - Y_1\hat{P}_1 - \frac{1}{\lambda_f}Y_{1f}\dot{\hat{P}}_1 - J^TF_I \\ &\quad - J^TG^{+}[Y_2\hat{P}_2 + \frac{1}{\lambda_f}Y_{2f}\dot{\hat{P}}_2]\end{aligned} \tag{45}$$

Replacing τ, τ_1 and τ_4 by their values from equations (35), (36) and (39), we get,

$$\begin{aligned}\frac{D+\lambda_f}{\lambda_f}(e_{p_1}) + J^TG^{+}\frac{D+\lambda_f}{\lambda_f}(e_{p_2}) &= \tau_1 + \tau_2 + \tau_3 - Y_1\hat{P}_1 - J^TG^{+}Y_2\hat{P}_2 - J^TF_I \\ &= (\tau_2 - Y_1\hat{P}_1 - J^TG^{+}Y_2\hat{P}_2) \\ &\quad + \tau_3 - J^T(e_F + K_I{\textstyle\int} e_F)\end{aligned} \tag{46}$$

Using (18), (19) and (15), we get,

$$\begin{aligned}-Y_1\hat{P}_1 - J^TG^{+}Y_2\hat{P}_2 &= -\hat{H}\ddot{q} - \hat{N}_1 - J^TG^{+}(\hat{S}\ddot{z} + \hat{N}_2) \\ &= -\hat{H}(J^{-1}G^T\ddot{z} - J^{-1}\dot{J}\dot{q}) - \hat{N}_1 - J^TG^{+}(\hat{S}\ddot{z} + \hat{N}_2) \\ &= -(\hat{H}J^{-1}G^T + J^TG^{+}\hat{S})\ddot{z} - (-\hat{H}J^{-1}\dot{J}\dot{q} + \hat{N}_1 + J^TG^{+}\hat{N}_2) \\ &= -\hat{H}_T\ddot{z} - (-\hat{H}J^{-1}\dot{J}\dot{q} + \hat{N}_1 + J^TG^{+}\hat{N}_2)\end{aligned} \tag{47}$$

If we combine (47) and (37), we get,

$$\tau_2 - Y_1\hat{P}_1 - J^TG^{+}Y_2\hat{P}_2 = -\hat{H}_T(\ddot{e} + K_v\dot{e} + K_pe) \tag{48}$$

Now combining equations (46) and (48), we have,

$$\begin{aligned}\frac{D+\lambda_f}{\lambda_f}(e_{p_1}) + J^TG^{+}\frac{D+\lambda_f}{\lambda_f}(e_{p_2}) &= -\hat{H}_T(\ddot{e} + K_v\dot{e} + K_pe) \\ &\quad + \tau_3 - J^T(e_F + K_I{\textstyle\int} e_F)\end{aligned} \tag{49}$$

Multiplying equation (49) by GJ^{-T}, we get,

$$\begin{aligned}GJ^{-T}\frac{D+\lambda_f}{\lambda_f}(e_{p_1}) + \frac{D+\lambda_f}{\lambda_f}(e_{p_2}) &= -\hat{A}(\ddot{e} + K_v\dot{e} + K_pe) \\ &\quad + GJ^{-T}\tau_3 - G(e_F + K_I{\textstyle\int} e_F)\end{aligned} \tag{50}$$

Now recall the fact that the internal grasping forces lie in the null space of G, thus we can conclude that

$$G(e_F + K_I{\textstyle\int} e_F) = 0 \tag{51}$$

Now substituting τ_3 by its value from equation (38), equation (50) becomes:

$$\begin{aligned}GJ^{-T}\frac{D+\lambda_f}{\lambda_f}(e_{p_1}) + \frac{D+\lambda_f}{\lambda_f}(e_{p_2}) &= -\hat{A}(\ddot{e} + K_v\dot{e} + K_pe) - \frac{1}{\lambda_f}G\dot{J}^{-T}e_{p_1} \\ &\quad - \frac{1}{\lambda_f}\hat{A}\dot{\hat{A}}^{-1}(GJ^{-T}e_{p_1} + e_{p_2})\end{aligned} \tag{52}$$

Equation (52) becomes:

$$\frac{D+\lambda_f}{\lambda_f}(GJ^{-T}e_{p_1} + e_{p_2}) = -\hat{A}(\ddot{e} + K_v\dot{e} + K_pe)$$

$$-\frac{1}{\lambda_f}\hat{A}\dot{\hat{A}}^{-1}(GJ^{-T}e_{p_1}+e_{p_2}) \tag{53}$$

Now multiply both sides of equation (53) by $\hat{A}^{-1}$, and use the identity given by Lemma 1, we get,

$$\frac{D+\lambda_f}{\lambda_f}[\hat{A}^{-1}(GJ^{-T}e_{p_1}+e_{p_2})]+(\ddot{e}+K_v\dot{e}+K_pe)=0 \tag{54}$$

If we let

$$u \triangleq -\frac{1}{\lambda_f}\hat{A}^{-1}(GJ^{-T}e_{p_1}+e_{p_2}) \tag{55}$$

Then equation (54) becomes

$$\ddot{e}+K_v\dot{e}+K_pe=(D+\lambda_f)u \tag{56}$$

Also it should be noted that $u \in L_2$ because of the following:
(1) The exact value of A, $(A = GJ^{-T}HJ^{-1}G^T+S)$, is a positive definite matrix because both H and S are positive definite matrices. We assumed that we are working in convex region of $\hat{P}_1$ and $\hat{P}_2$ such that $\hat{H}$ and $\hat{S}$ are positive definite, which implies that $\hat{A}$ is positive definite and thus $\hat{A}^{-1}$ is bounded.
(2) GJ^{-T} which is a function of the kinematic parameters of the manipulators (which are assumed to be known) is bounded.
(3) In theorem 1 we proved that $e_{p_1} \in L_2$ and $e_{p_2} \in L_2$.
Hence using Lemma 2 and the facts that $u \in L_2$ and $\hat{A}^{-1}$ is bounded, we can conclude that $e \longrightarrow 0$ as $t \longrightarrow \infty$.

In the following we will prove that $u \longrightarrow 0$ as $t \longrightarrow \infty$ and therefore $\dot{e} \longrightarrow 0$ as $t \longrightarrow \infty$. Middleton and Goodwin [5], proved that the regressor matrix Y_f and its derivative $\dot{Y}_f$ are bounded for rigid link manipulator systems. Therefore Y_{1f} and Y_{2f} will also satisfy the same property because they are also represented by the same rigid mechanical dynamics. Hence using the same arguments as Middleton and Goodwin (we omit the proofs here, see page 13-14 of [5]), it can be shown that Y_{1f}, $\dot{Y}_{1f}$, Y_{2f} and $\dot{Y}_{2f}$ are bounded. Thus e_{p_1} is bounded because $e_{p_1}=-Y_{1f}\tilde{P}_1$. Now $\dot{\hat{P}}_1=\Gamma_1Y_{1f}^Te_{p_1}$ is bounded because Γ_1, Y_{1f} and e_{p_1} are bounded. Also, $\dot{e}_{p_1}=-\dot{Y}_{1f}\tilde{P}_1-Y_{1f}\dot{\hat{P}}_1$ is bounded. Therefore we can conclude e_{p_1} is uniformly continuous. Since e_{p_1} is uniformly continuous and $e_{p_1} \in L_2$, we have $e_{p_1} \longrightarrow 0$ as $t \longrightarrow \infty$. Similarly e_{p_2} is bounded because $e_{p_2}=-Y_{2f}\tilde{P}_2$. Now $\dot{\hat{P}}_2=\Gamma_2Y_{2f}^Te_{p_2}$ is bounded because Γ_2, Y_{2f} and e_{p_2} are bounded. Also, $\dot{e}_{p_2}=-\dot{Y}_{2f}\tilde{P}_2-Y_{2f}\hat{P}_2$ is bounded. Therefore we can conclude that e_{p_2} is uniformly continuous. Since e_{p_2} is uniformly continuous and $e_{p_2} \in L_2$, we have $e_{p_2} \longrightarrow 0$ as $t \longrightarrow \infty$. Hence $u=-\frac{1}{\lambda_f}\hat{A}^{-1}(GJ^{-T}e_{p_1}+e_{p_2})$ tends to zero as t tends to infinity. Finally we can conclude from Lemma 2 that $\dot{e} \longrightarrow 0$ *as* $t \longrightarrow \infty$. Thus we can conclude that $\hat{P}_1$, $\hat{P}_2$, $\dot{\hat{P}}_1$, $\dot{\hat{P}}_2$ are bounded and also e_{p_1} , e_{p_2}, e, $\dot{e}$ tend to zero as t tends to infinity.
To prove that e_F tends to zero, we should note that equation (49) can be written as:

$$\frac{D+\lambda_f}{\lambda_f}(e_{p_1})+J^TG^+\frac{D+\lambda_f}{\lambda_f}(e_{p_2})=-\hat{H}_T(\ddot{e}+K_v\dot{e}+K_pe)-\frac{1}{\lambda_f}J^T\dot{J}^{-T}e_{p_1}$$
$$-\frac{1}{\lambda_f}J^TG^+\hat{A}\dot{\hat{A}}^{-1}(GJ^{-T}e_{p_1}+e_{p_2})-J^T(e_F+K_I\textstyle\int e_F) \tag{57}$$

Now because e_{p_1}, e_{p_2}, e and $\dot{e}$ tend to zero as t tends to infinity we have

$$J^T(e_F+K_I\textstyle\int e_F)\longrightarrow 0 \tag{58}$$

However J^T is not singular, thus we have

$$e_F + K_I \int e_F \longrightarrow 0 \tag{59}$$

Finally because K_I is a positive semidefinite matrix, we have $e_F \longrightarrow 0$ as $t \longrightarrow \infty$.

This concludes the proof of theorem 2.

5. Conclusion

In this paper, we proposed an adaptive control scheme for the multi-robot system during cooperative motion. The proposed controller takes into account the dynamics of the object and the dynamics of the manipulators. The linearity of the dynamics of both the robots and the load with respect to the parameters, were exploited during the derivation of the controller. The dynamics of both the robots and the load were filtered so that the control law does not require the measurements of the joint accelerations and the load acceleration. The proposed controller guarantees convergence of the actual position to the desired one and the convergence of the internal grasping forces to the desired internal grasping forces. Least-squares estimation was used to estimate the parameters. Future work on using different methods for parameter estimation, and robustness of the least-squares estimator is underway.

References

[1] D. Bayard, and J. Wen, New Class of Control Laws for Robotic Manipulators: Part 2. Adaptive case, Int. J. Control, Vol. 47, No. 5, 1985, pp. 1387-1406.

[2] J. Craig, P. Hsu, and S. Sastry, Adaptive Control of Mechanical Manipulators, Proceedings of the 1986 IEEE International Conference on Robotics and Automation, San Francisco, CA, 1986, pp. 190-195.

[3] C. Desoer and M. Vidyasagar, Feedback Systems: Input-Output Properties, Academic Press, New York, 1975.

[4] P. Hsu, Z. Li, and S. Sastry, On Grasping and Coordinated Manipulation by a Multifingered Robot Hand, Proceedings of the 1988 IEEE International Conference on Robotics and Automation, Philadelphia, Pennsylvania, 1988, pp. 384-389.

[5] R. Middleton and G. Goodwin, Adaptive Computed Torque Control for Rigid Link Manipulators, Systems & Control Letters Vol. 10, 1988, pp. 9-16.

[6] R. Ortega and M. Spong, Adaptive Motion Control of Rigid Robots: a Tutorial Automatica, Vol. 25, No 6, 1989, pp. 877-888.

[7] N. Sadegh, and R. Horowitz, Stability and Robustness Analysis of a Class of Adaptive Controllers for Robotic Manipulators, Int. J. of Robotics Research, Vol. 9, No 3, 1990, pp. 74-92.

[8] J-J. Slotine and W. Li, Adaptive Manipulator Control: A Case Study, IEEE Trans. Automatic Control, Vol. AC-33, No. 11, 1988, pp. 995-1003.

[9] J-J. Slotine and W. Li, Applied Nonlinear Control, Prentice Hall, New Jersy, 1991.

[10] M. Walker, D. Kim and J. Dionise, Adaptive Coordinated Motion Control of Two Manipulator Arms, Proceedings of the 1989 IEEE International Conference on Robotics and Automation, Scottsdale, Arizona, 1989, pp 1084-1090.

[11] M. Zribi and S. Ahmad, Robust Adaptive Control of Multiple Robots in Cooperative Motion Using σ Modification, Proceedings of the 1991 IEEE International Conference on Robotics and Automation, Sacramento, CA, 1991.

INDEX OF AUTHORS

Abrahamson, K. 642
Ahmad, M. 733
Ahmad, S. 1127, 1137
Aicardi, M. 342
Aksu, H. A. 977
Anantharam, V. 302
Andrews, M. G. 831
Antsaklis, P. J. 693
Aravena, J. L. 737, 929
Arora, N. 958
Azemi, A. 729

Bambos, N. 310
Bauer, P. H. 939, 948
Benromdhane, S. 735
Bensoussan, D. 84, 1114
Bertsekas, D. P. 989
Bhatia, S. 488
Bhattacharya, P. P. 312
Bhattacharyya, N. 929
Bingulac, S. 448
Birk, Y. 322
Blair, W. D. 559, 561
Blask, S. G. 191
Bolla, R. 332
Borth, D. E. 11
Bottura, C. P. 703
Brady, D. 390
Brassil, J. 571
Burman, S. 1083

Chai, Z. 179
Chandrasekharan, N. 475
Chen, B. M. 50, 74
Chen, C.-C. 760
Chen, G.-Q. 751
Chen, H. 1083
Chen, W. Z. 211
Chen, Y.-W. 979
Cheng, Y. C. 1013
Chiang, H.-D. 760
Chung, H. W. 632
Cioffi, J. M. 1
Clarkson, P. M. 127, 555
Coffey, J. T. 790
Corless, M. 885
Coxson, G. E. 105
Cruz, R. 571
Cucka, P. 222

Da, D. 885
Dahleh, M. 866
Das, S. 958
Datta, A. K. 672
Davoli, F. 332
DeCarlo, R. A. 169, 430
DeMarco, C. L. 105
Dekel, E. 508
Delingette, H. 181
Deng, W. 856
Dhall, S. K. 484, 486
DiFebbraro, A. 332
Dimino, S. A. 925
Dube, D. 613

El-Khazali, R. 430
Elrad, T. 613
Eslami, M. 107

Fabien, B. C. 60
Filho, J. T. C. 703
Firby, R. J. 233
Flamm, D. S. 72, 463
Flatebo, M. 672
Forest, T. M. 557
Fu, J. 922
Furuta, R. 272

Gajic, Z. 440, 455
Gallager, R. G. 591
Garg, V. 705
Georgiadis, L. 312
Ghosh, S. 662, 674
Grenander, U. 563
Grey, W. 1053
Gupta, A. 243

Hammami, M. 84
Han, J.-Y. 751
Hebert, M. 181
Hegde, M. 737, 739, 958
Heinen, J. A. 925, 927
Herve, J.-Y. 222
Hirschberg, D. S. 488
Hmam, H. M. 1117
Hodel, A. S. 473
Holtzman, J. M. 352
Hood, S. 1053
Hou, J. 731

Hu, J. 508

Ichisugi, Y. 263
Ikeuchi, K. 181
Iyengar, S. S. 856

Jagannathan, S. 253
Janardan, R. 1003
Jordan, S. 284
Juang, T. T.-Y. 652

Kabal, P. 779, 798
Kailath, T. 159
Kak, A. C. 191
Karaata, M. H. 674
Karkare, A. 642
Kashyap, R. L. 534
Kattepur, S. M. 534
Kesidis, G. 581
Kessler, I. 622
Ketchum, J. 40
Ketel, M. 543
Khandani, A. K. 779, 798
Khargonekar, P. P. 905
Kiely, A. B. 790
Klipec, K. 72
Ko, S.-J. 557
Kogan, J. 98
Konstantopoulos, P. 302
Korde, U. A. 211
Kriegman, D. J. 201
Krishna, A. 622
Krishnamoorthy, M. S. 498
Kumar, S.P.R. 632
Kumar, R. 705
Kumar, V. 243
Kurz, L. 543

Lakshmivarahan, S. 484, 486, 632
Lam, A. W. 370
Lau, F. C. M. 1103, 1105
Lau, S. W. 1103
Lawrence, D. A. 1117
Lee, D. T. 831
Lee, F. 841
Lee, S. C. 979, 1013
Lee, S. 613
Lehnert, J. S. 362
Lei, X. 751
Li, Q. 31
Li, S. 1063
Liang, Y. D. 484, 486
Lin, P.-Y. F. 1033
Lin, X. 662
Lin, Z.-L. 74
Liu, Y.-J. 40, 769
Lopez, M. 1003
Low, S. 593
Lu, M. 179
Luo, S. 1127

MacDonald, J. T. 715
Madhow, U. 380
Matsuoka, S. 263
Mavronicolas, M. 683
McKinley, M. S. 410
Mersereau, R. M. 139
Michel, A. N. 741
Miller, M. I. 563
Minciardi, R. 342
Morris, J. M. 553

Nagpal, K. M. 905
Nakajima, K. 1033, 1081
Nandakumar, G. 739
Naraghi-Pour, M. 737, 739, 958
Narasimhan, J. 1081
Nassimi, D. 845
Niederjohn, R. J. 925, 927
Nowack, J. M. 11
Null, R. E. 1061

Olariu, S. 811, 821
Oommen, B. J. 518
O'Sullivan, J. A. 524
Ozbay, H. 912

Packard, A. 875
Pajunen, G. A. 420
Pal, D. 159
Pandelis, D. G. 294
Pandey, P. 875
Parekh, A. K. 591
Parker, J. D. 561
Pawlak, M. 727
Pesenti, R. 342
Petersen, I. R. 885
Petkovski, D. 440
Potter, L. C. 149
Pour, M. N. 737, 739, 958

Qian, S.-E. 553
Qureshi, M. T. 455

Ramamurthy, B. 498
Rao, T. S. 139

Rasky, P. D. 11
Rencher, M. R. 1071
Rhee, C. 484, 486
Rim, C. S. 1081
Rimoldi, B. 31
Romain, P. F. 809
Rotea, M. 885
Roth, D. 683

Saberi, A. 50, 74
Salam, F. M. A. 731, 733, 735
Salehi, M. 809, 977
Sarrafzadeh, M. 1023
Sasaki, G. 603
Scherson, I. D. 488
Schwing, J. L. 811, 821
Shamma, J. S. 895
Sharma, R. 222
Sherwani, N. 1083
Simpson, F. 352
Skaar, S. B. 211
Skataric, D. 440
Skelton, R. E. 717
Sridhar, R. 475
Srivastava, A. 563
Stark, W. E. 410
Stiver, J. A. 693
Stoorvogel, A. A. 50
Stotts, P. D. 272
Subrahmanyam, M. B. 70
Sun, W. 905
Suzuki, I. 1043
Svoren, T. J. 925, 927

Tadmor, G. 88
Tang, S. 922
Tantaratana, S. 370
Taylor, C. J. 201
Taylor, S. 282
Teneketzis, D. 294
Tesi, A. 866
Tiller, J. 473
Tsoucas, P. 312

Vaidyanathan, R. 1093
Valiveti, R. S. 518
VanLandingham, H. F. 448
Van Oyen, M. P. 294
Varaiya, P. 593
Varanasi, M. K. 400
Vasudevan, S. 400
Venkatesan, S. 652
Verma, M. 88
Verriest, E. I. 420
Vicino, A. 866

Wallace, M. S 21, 40
Wallace, R. B. 927
Walrand, J. 581
Wang, J. 939, 948
Wasserman, K. 310
Watanabe, T. 263
Wen, Z. 847
Wicks, M. 169
Williamson, G. A. 127, 555, 715
Wong, P. W. 129
Wu, J. 751
Wu, X. 968

Xu, C. Z. 1105

Yamashita, M. 1043
Yaz, E. 717, 729
Yeap, K.-H. 1023
Yeh, C.-C. 760
Yen, G. 741
Yin, G. 117
Yonezawa, A. 263
Yuan, J.-L. 760

Zhang, J. 811, 821
Zhang, K. 922
Zhang, Y. 843
Zhao, L. 117
Zheng, S.-Q. 841, 843
Zhu, W. 179
Zou, M. 179
Zribi, M. 1137
Zvonar, Z. 390

UNIVERSITY OF ILLINOIS-URBANA
3 0112 045661649